VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

AUTOMATICKÁ PODPORA TVORBY DOKUMENTACE PROJEKTŮ

AUTOMATIC SUPPORT OF DOCUMENTATION OF PROJECTS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **DANIEL VOSÁHLO**
AUTHOR

VEDOUCÍ PRÁCE **Ing. JAROSLAV DYTRYCH**
SUPERVISOR

BRNO 2017

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav počítačové grafiky a multimédií Akademický rok 2016/2017

Zadání bakalářské práce

Řešitel: **Vosáhlo Daniel**
Obor: Informační technologie
Téma: **Automatická podpora tvorby dokumentace projektů**
Automatic Support of Documentation of Projects
Kategorie: Web

Pokyny:

1. Seznamte se s jazyky a prostředky pro tvorbu webových informačních systémů.
2. Prostudujte MediaWiki, informační systém výzkumné skupiny znalostních technologií (KNOTIS) a současné metody generování hlaviček stránek projektů.
3. Navrhněte rozšíření KNOTIS a další potřebné skripty pro automatické generování dokumentace projektů a skupin projektů. Zaměřte se při tom na detailní nastavení možností generování pro jednotlivé projekty a vhodné využití maximálního množství dostupných informací.
4. Implementujte navržené řešení.
5. Zhodnoťte dosažené výsledky, srovnejte zvolené přístupy s alternativními metodami a navrhněte další možná rozšíření a vylepšení do budoucna.

Literatura:

- Dle doporučení vedoucího.

Pro udělení zápočtu za první semestr je požadováno:

- body 1, 2 a 3

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Dytrych Jaroslav, Ing.**, UPGM FIT VUT
Datum zadání: 1. listopadu 2016
Datum odevzdání: 17. května 2017

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav počítačové grafiky a multimédií
612 66 Brno, Božetěchova 2

doc. Dr. Ing. Jan Černocký
vedoucí ústavu

Abstrakt

Cílem této práce je vytvořit program, který bude automaticky vytvářet zvolenou úroveň dokumentace z informačního systému pro podporu vědy a výzkumu a umožňovat uživatelský zásah do ní. Zvolený problém jsem vyřešil vytvořením programu v jazyce PHP, který využívá soukromé Wiki výzkumné skupiny KNOT a vytváří speciální hlavičky na určených stránkách. V práci jsem vytvořil detailní návrh postupu generování a kontroly těchto hlaviček, který dokáže rozeznat uživatelský text v hlavičce a pracovat s ním. Podle tohoto návrhu jsem implementoval program, který slouží jako platforma a jeho funkčnost lze rozšířit moduly. Program periodicky kontroluje údaje v informačním systému, ale i obsah hlaviček, jestli v nich uživatel manuálně nezměnil obsah. Pokud se v hlavičce nic nezměnilo, tak tyto hlavičky dokáže přeskočit. Toto zkrátilo dobu kontroly více než dvojnásobně a umožnilo snadné přidávání funkcionality. Vytvořené řešení zabraňuje ztrátě uživatelských dat a velmi urychluje a zefektivňuje kontrolu celé dokumentace.

Abstract

The goal of this thesis is to create a program which will automatically create selected level of documentation from an information system for supporting science and research and enables user intervention. I solved the selected problem with creating a program in the PHP language which uses a private MediaWIKi which belongs to the KNOT research group and creates special headers on certain pages. In this work I created a detailed proposal of the procedure that generates and checks these headers and it can tell apart user text and work with him. According to this design, I implemented a program that serves as a platform and its functionality can be expanded by modules. The program periodically checks the information in the information system, as well as the contents of the headers, if the user did not manually change the content and if nothing else changed in the header, these headers can be skipped. This shortened the program running time more than twice, and has made it easy to add new functionalities. This solution prevents the loss of user data, greatly speeds up the control of the entire documentation and makes it more efficient.

Klíčová slova

Informační systém pro podporu vědy a výzkumu, automatické generování dokumentace, MediaWiki bot, PHP, rozpoznání uživatelského obsahu

Keywords

Information system for supporting science and research, automatic documentation of projects, MediaWiki bot, PHP, user content recognition

Citace

VOSÁHLO, Daniel. *Automatická podpora tvorby dokumentace projektů.* Brno, 2017. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Jaroslav Dytrych

Automatická podpora tvorby dokumentace projektů

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Jaroslava Dytrycha. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Daniel Vosáhlo
16. května 2017

Poděkování

Rád bych poděkoval svému vedoucímu Ing. Jaroslavu Dytrychovi za vedení práce, odborné rady a vřelý přístup.

Obsah

Kapitola 1

Úvod

Dnešní svět je založen na technice, vědě a pokroku. Stavíme ty nejvyšší budovy, ty nejdelší tunely, nejtenčí mobilní telefony a další věci, o kterých se mohlo našim předkům jenom zdát. Jak se postupy zesložiťují a zasahují do více oborů současně, tak roste potřeba po kvalitním dokumentování použitých technologií a postupů řešení. Toto ještě více platí o výzkumných skupinách působících na univerzitách, které vždy byly průkopníky a hybateli vědy. Tyto skupiny mohou mít od pár až po několik stovek členů a mohou pracovat až na desítkách projektů a výzkumů současně. Jelikož tyto vědecké skupiny většinou pracují na postupech, které před nimi nikdo nikdy nepoužil, nebo nedokázal použít, tak jejich potřeba vést dobrou dokumentaci je kritická.

Tato práce pojednává o způsobu, jak je možné efektivně automaticky generovat a kontrolovat aktuálnost dokumentace projektů, které jsou řešeny univerzitní výzkumnou skupinou, při zachování možnosti zásahu uživatele nebo možnosti kompletní uživatelské dokumentace. Výsledkem této práce je program, který implementuje navržený způsob generování dokumentace a je schopný rozlišit vygenerovanou dokumentaci od uživatelské.

Osobně jsem velikým fanouškem techniky. Rád se dozvídám nové věci, zjišťuji jak tyto věci fungují a nevyhýbám se výzvám. Také rád dělám věci, které se využívají a jsou užitečné dalším lidem. Proto mě zaujalo toto téma, které představovalo možnost vytvořit něco, co bude užitečné pro ostatní, a také představovalo výzvu.

Ve 2. kapitole této práce se dozvíte něco o možnostech automatického generování dokumentace z informačního systému a základní informace o výzkumné skupině, pro kterou je program určen. Kapitola 3 popisuje použité technologie, které umožnily vytvoření tohoto programu. Dále budou v kapitole 4 popsána data reprezentující obsah, ke kterému má být vygenerována dokumentace. Tato část popisuje také trend, jakou rychlostí tato data vznikají. Kapitola 5 popisuje současné řešení a obsahuje návrh nového řešení. Tato sekce podrobně vysvětluje všechny použité postupy a je kritická k pochopení celé práce. Kapitola 6 popisuje způsob, jakým byl z návrhu vytvořen funkční program. Na tuto kapitolu navazuje kapitola 7, která se zabývá testováním vytvořeného programu. A na závěr následuje shrnutí celé práce a zhodnocení výsledků obsažené v kapitole 8.

Kapitola 2

Automatické generování dokumentace z informačního systému pro podporu vědy a výzkumu

Výzkumné skupiny jsou často tvořeny mnoha členy a zabývají se širokou škálou projektů a výzkumů. Je nutné dobře evidovat informace, postupy a výsledky všech výzkumů, stejně jako je nutné evidovat zadané úkoly a vykonanou práci. Pro správu všech těchto dat se používají informační systémy. Někdy ovšem samotný informační systém nestačí, například pokud je nutné mít složitější dokumentaci k řešeným projektům nebo je počet projektů tak vysoký, že uložení dokumentace uvnitř informačního systému by bylo nepraktické.

2.1 Systém pro podporu vědy a výzkumu KNOTIS

Výzkumná skupina znalostních technologií (dále jen KNOT) je jedna z nejmladších výzkumných skupin na Fakultě informačních technologií Vysokého učení technického v Brně. Zaměřuje se hlavně na zpracování přirozeného jazyka pomocí automatizovaných nástrojů, následné bezprostřední začlenění do uživatelských rozhraní a na interakcí člověka s počítačem.

KNOT se zabývá širokou škálou projektů a na jejich řešení se podílejí především studenti pod vedením odborných pracovníků. Díky tomuto přístupu je nutné evidovat úkoly zadané jednotlivým řešitelům, prostředky, které mají k dispozici, a práci, kterou vykonali. Také je nutné uschovávat projekty i po skončení jejich řešení, aby případní jiní zájemci mohli v projektu pokračovat a měli přehled o práci svých předchůdců.

Pro tyto potřeby vyvinula skupina KNOT vlastní informační systém nazvaný KNOTIS. V KNOTIS jsou vedeny základní informace o skupinách projektů a projektech, jako je třeba datum zahájení, ukončení, informace o řešitelích, atd. Také jsou v něm ulo-

ženy úkoly pro jednotlivé řešitele, přehled změn stavů úkolů a odeslané pracovní výkazy. Všechny tyto informace jsou v něm trvale uloženy až do doby, než je administrátor ručně odstraní. Díky tomuto přístupu a velkému počtu projektů není praktické do něj ukládat obsáhlou dokumentaci. Proto byl pro tvorbu a aktualizaci dokumentace vytvořen jiný, alternativní způsob uložení.

2.2 Generování dokumentace pomocí speciálních hlaviček

Jedním z alternativních způsobů generování a udržování aktuální dokumentace je automatické generování pouze základních informací o dokumentovaném subjektu, které se vezmou z informačního systému, a zbytek dokumentace vytvoří a udržuje řešitel sám.

KNOT používá tento způsob dokumentace, kdy se z KNOTIS načtou potřebné informace, upraví se do požadovaného formátu a ve formě speciální hlavičky se uloží jako záhlaví WIKI stránky do soukromé WIKI výzkumné skupiny KNOT (dále jen KNOT WIKI). KNOT WIKI působí jako sklad dokumentace s rozsáhlými možnostmi editace, správy a vedení historie jednotlivých stránek.

V současné době výzkumná skupina KNOT používá pro automatické generování a kontrolu dokumentace program, který byl výsledkem diplomové práce Aleše Vavřínka [11]. Tento program byl vytvořen v roce 2011 v době, kdy byl informační systém KNOTIS ve vývoji a nebyl ještě zcela dokončen. Program vytváří speciální hlavičky, jak bylo popsáno výše, umí kontrolovat aktuálnost skupin projektů, projektů, úkolů a výkazů řešitelů. Reálně se dnes používá pouze kontrola skupin projektů a projektů. Toto se dá nastavit v informačním systému KNOTIS, kde se nachází speciální stránky s nastavením generování dokumentace.

Kapitola 3

Použité technologie

Pro vytvoření této práce bylo použito hned několik technologií a postupů. Ty nejdůležitější budou popsány dále v textu.

3.1 PHP

Pro implementaci mého programu jsem zvolil jazyk PHP. PHP je všestranný a rychlý skriptovací jazyk, který je v současné době používán převážně pro tvorbu dynamických webů. Dá se také využít pro tvorbu klasických desktopových nebo konzolových aplikací. Skript v PHP je možné volat z příkazového řádku, nebo pomocí webového serveru, kde jsou skripty uloženy a zpracovány na straně serveru [6].

PHP jsem si pro implementaci vybral z několika důvodů. Tím největším bylo to, že tento jazyk dobře znám a informační systém KNOTIS je v něm také napsán. Dalším důvodem bylo, že je k dispozici nová verze tohoto jazyka, která značně vylepšuje jeho výkon. Neposledním důvodem je fakt, že tato verze je přítomna jako základní verze skoro ve všech nových operačních systémech.

Touto novou verzí je PHP verze 7, která běží na úplně novém jádře, které poskytuje spoustu nových funkcionalit a nabízí razantně zvýšený výkon a sníženou paměťovou náročnost [3]. Také umožňuje deklaraci typu předávaného argumentu, typu návratové hodnoty funkce, přidává lepší práci s výjimkami a kritickými chybami [8]. Všechny tyto vlastnosti jsou užitečné pro vytvoření programu, který načítá velké množství dat, komunikuje s API[1] a kontroluje stovky stránek.

3.2 phpDocumentor

PhpDocumentor zkráceně phpDoc je volně dostupný program, který slouží pro automatické generování dokumentace souborů v jazyce PHP [2]. Tato dokumentace je generována v podobě webových stránek. Uživatel má na výběr z několika předloh a vzhledů těchto stránek. Pro jeho použití je potřeba na začátek každého souboru, třídy, funkce

[1] API – Application Programming Interface (Rozhraní pro programování aplikací)

nebo metody vložit speciální komentář. Tento komentář může obsahovat popis, vstupní proměnné, návratový typ, informace o autorovi, atd.

3.3 cURL

Libcurl je knihovna pro jazyk PHP, která byla vytvořena Danielem Stenbergem a slouží pro příjem a posílání požadavků na server. *Knihovna umí využívat celou řádu protokolů, jako například http, https, ftp, gopher, telnet, dict, file a ldap. Také podporuje certifikáty HTTPS, posílání formulářů, práci s dočasnými soubory a přihlášení uživatele ke službě* [5]. Díky této knihovně může program posílat požadavky na server a zpracovávat jeho odpovědi.

3.4 XxHash

Pro zrychlení a zefektivnění kontroly jsem použil postup, při kterém jsem vytvořil matematický otisk celé hlavičky (dále jen hash), nebo její části a porovnal s hashem, který byl uložen v databázi z předešlé kontroly programu. Tento postup bude podrobněji vysvětlen v kapitole 5.2.

Pro vytváření hashů jsem zvolil knihovnu Xxhash, kterou pro PHP7 vytvořil Craig R. Megasaxon a která implementuje algoritmus xxHash vytvořený Yannem Colletem. *XxHash je velmi rychlý ne-kryptografický hashovací algoritmus pracující rychlostí pamětí RAM* [4]. Tato knihovna poskytuje srovnatelnou, nebo lepší kvalitu otisku, co se týče kolizí a je nesrovnatelně rychlejší než jakákoli zabudovaná hashovací funkce v jazyce PHP.

3.5 Databázový systém MySQL

Informační systém KNOTIS a KNOT WIKI používají pro uložení svých dat databázový systém MySQL. MySQL je jedním z nejpopulárnějších multiplatformních volně dostupných databázových systémů používajících jazyk SQL. Je vyvíjen firmou Oracle a je volně dostupný pod licencí GNU[2], nebo také pod placenou komerční licencí.

MySQL používá relační databázový model, což znamená, že data jsou organizována v relacích. *Relace je základní abstraktní pojem relačního modelu a jeho znázornění se nazývá tabulka* [14]. MySQL neukládá data do jednoho velkého souboru, ale ukládá je do menších souborů, většinou je jedna tabulka uložena v jednom souboru. Díky tomuto může MySQL dosahovat velmi dobré výkonnosti při práci s objemnými daty. MySQL může být nasazena na osobním počítači či serveru [12]. Tato vlastnost je dobrá, pokud uživatel potřebuje pracovat s kopií databáze na svém počítači.

Databáze informačního sytému KNOTIS je velice rozsáhlá a obsahuje tabulky, které mají i několik miliónů řádků. Proto je při návrhu a implementaci řešení důležité klást důraz na optimalizaci dotazů i jejich počtu [9].

[2]GNU – General Public License (Obecná veřejná licence) http://www.gnugpl.cz/v3/

3.6 Současná verze systému

V současné době běží informační systém KNOTIS a všechny ostatní aplikace skupiny KNOT na operačním systému Ubuntu ve verzi 14.4. Jádro tohoto systému je již zastaralé a plánuje se aktualizace na novější verzi systému Ubuntu. S touto aktualizací vyvstala spousta problémů, protože většina aplikací běží na starých verzích balíků a knihoven. Kvůli tomuto faktu bylo potřeba upravit informační systém KNOTIS, zejména jeho rozhraní v PHP, aby bylo kompatibilní s novou verzí systému. Obsahovalo totiž funkce pro práci s databází, které se již v nové verzi nevyskytují, a funkce pro práci s regulárními výrazy, které již také byly vyřazeny.

3.7 Media Wiki

KNOT WIKI běží na systému zvaném Media Wiki. Media Wiki je serverově orientovaný WIKI systém, který je volně k dispozici pod licencí GNU. Tento systém je velmi výkonný a škálovatelný podle potřeb a rozsahu jeho využití. Byl vytvořen za kombinace jazyka PHP, který obstarává zpracování vstupů a generování stránek, a databáze MySQL, která slouží k uložení dat. Media Wiki také používá svou vlastní syntaxi pro formátování dat na stránce nazvanou WIKI MarkUp.

WIKI MarkUp je založen na jazyce HTML[3], a pokud uživatel chce formátovat svůj text, tak musí tuto syntaxi využít. WIKI pak tuto syntaxi automaticky převede do HTML a zobrazí. Díky tomuto přístupu může i uživatel neznalý HTML vytvářet na své WIKI stránce tabulky, odstavce, formátovat písmo a jinak upravovat stránku [1].

Media Wiki také uchovává historii všech změn na stránce a informace o tom, kdo stránku editoval. Také nabízí širokou škálu oprávnění pro uživatele a jejich správu.

Výzkumná skupina KNOT v současné době používá Media Wiki ve verzi 1.23, která začíná být zastaralá. V nových verzích se změnil přístup k programům, které pracují s Media Wiki a automaticky na ní upravují data. Tyto programy se nazývají boti. Také se změnily některé funkce a parametry rozhraní pro komunikaci s Media Wiki (dále jen API) a přibylo silné doporučení k požadavkům bota přidávat speciální hlavičku protokolu HTTP[4].

Nyní Media Wiki podporuje možnost vytvořit pro bota speciální identifikátor a heslo. Díky tomuto se bot již nemusí přihlašovat s údaji jeho majitele nebo se botovi nemusí vytvářet speciální účet. Botovi se také dají nastavit různá oprávnění pro práci s Wiki a různé IP adresy, ze kterých se může přihlašovat [10].

Media Wiki API je realizováno jako webové. To znamená, že existuje speciální URL[5], na kterou se posílají požadavky HTTP typu GET. To znamená, že jednotlivé parametry

[3]HTML – HyperText Markup Language (Hypertextový značkovací jazyk) https://html.spec.whatwg.org/

[4]HTTP – Hypertext Transfer Protocol (Internetový protokol určený pro výměnu hypertextových dokumentů) https://www.w3.org/Protocols/

[5]URL – Uniform Resource Locator (Jednoznačné umístění zdroje) – je to řetězec, který slouží pro jednoznačnou specifikaci umístění zdroje na Internetu.

požadavky jsou součástí adresy. Zde je příklad požadavku, který vrátí obsah stránky *Main Page* z anglické verze Wikipedie:

```
https://en.wikipedia.org/w/api.phpaction=query&titles=Main%20Page
&prop=revisions&rvprop=content&format=php
```

WIKI tento požadavek zpracuje a pošle odpověď ve formátu, který v požadavku definujeme. V předešlém příkladě by API vrátilo odpověď ve formátu PHP. V nových verzích Media Wiki bylo mnoho již zastaralých funkcí odstraněno a změnil se i přístup pro získání obsahu stránky [13].

I kvůli výše zmíněným skutečnostem jsem se po dohodě se svým vedoucím rozhodl implementovat svou práci pro nejnovější verzi systému Media Wiki a aktualizovat verzi, na které běží KNOT WIKI.

Kapitola 4

Data

Jak bylo popsáno výše, program si bere data z různých částí informačního systému KNOTIS. Některá data jsou vygenerována automaticky a některá vložena nebo upravena uživatelem. Tomu také odpovídá různý formát získaných dat.

Data obsahují velkou škálu informací, jako například URL, názvy projektů nebo skupin projektů, jména, přijení a tituly uživatelů a příznaky, jestli je uživatel vedoucím nějakého projektu či skupiny projektů. Také jsou v datech uloženy informace o důležitých datech, jako jsou začátky a konce řešení projektů. Dále jsou uloženy úkoly pro řešitele s historii jejich stavů, pracovní výkazy řešitelů, informace o přidělených prostředcích, oprávnění pro jejich používání a mnoho dalších informací. Na přiloženém paměťovém médiu v originálu této práce je pak uloženo detailní schéma databáze informačního systému KNOTIS.

K této rozmanitosti dat tu je i občasná zmatečnost informačního systému KNOTIS, kdy některé názvy stavů jsou uloženy v databázi a některé jsou uloženy v konfiguračním souboru.

Dalším příkladem zmatečnosti dat je nejednotnost uložených URL Wiki stránek. Automaticky vygenerované adresy ve formuláři v sobě obsahují jméno nebo zkratku daného projektu, či skupiny projektů. Tady nastává komplikace, že uživatel může změnit adresu v KNOTIS ručně a nemusí dodržet velká a malá písmena. Toto může způsobit problémy například v procesu, kdy se porovnává URL s adresami, které jsou uloženy v KNOTIS jako adresy, na kterých se nemá nic generovat, ani k nim přistupovat (dále jen WIKI výjimky). Tyto adresy mají jiný formát než adresy pro projekty nebo skupiny a všechny tyto adresy může uživatel upravit.

Trochu vzácnějším příkladem nejednotnosti URL je jejich kódování. Drtivá většina URL uložených v KNOTIS je ve standardním formátu, kromě pár výjimek, které byly převzaty ještě z předešlého informačního systému a obsahují české znaky uložené v kódování URL, jako například adresa pro projekt Morfologický slovník a morfologický analyzátor pro češtinu:

```
https://.../Morfologick%C3%BD_slovn%C3%ADk_a_morfologick%C3%BD...
```

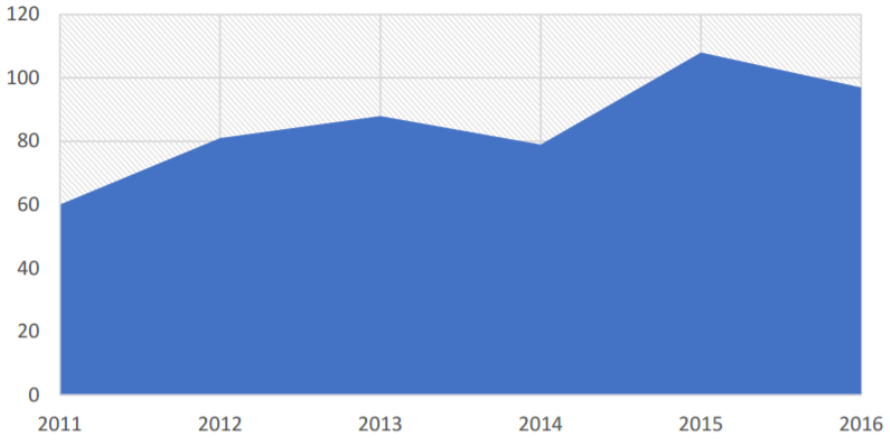
Statistika

Jak bylo zmíněno v kapitole 2.1, v informačním systému KNOTIS jsou uloženy i projekty, jejichž řešení již skončilo, což znamená, že je v něm uložena velká spousta dat, jak je vidět v tabulce 4.1. V tabulce je také vidět, že více projektů může sdílet 1 Wiki stránku nebo projekt může mít vypnuté generování. To znamená, že na 1 stránce může být 0..N hlaviček.

Název	Počet v KNOTIS	Má zapnuté generování	Unikátní Wiki adresy
Skupiny projektů	10	10	10
Projekty	619	502	496

Tabulka 4.1: Počet projektů a skupin projektů, které jsou uložené v informačním systému KNOTIS. Tabulka také ukazuje počet vygenerovaných hlaviček a unikátních WIKI stránek, které se mají kontrolovat.

Na obrázku 4.1 je vidět stoupající tendence zadávání projektů v systému KNOTIS. Díky této statistice můžeme odhadnout, jakým tempem bude přibývat projektů a jaké nároky budou v budoucnu kladeny na automatické generování dokumentace a na dokumentaci samotnou.



Obrázek 4.1: Ukázka počtu zadaných projektů za rok do informačního systému KNOTIS od roku 2011 do roku 2016.

Kapitola 5

Současný stav generování dokumentace a návrh nového systému na generování dokumentace

Tato kapitola popisuje problémy současného systému pro generování dokumentace, důvody, které vedly k jeho nahrazení, a návrh nového systému.

5.1 Problémy současného generování hlaviček

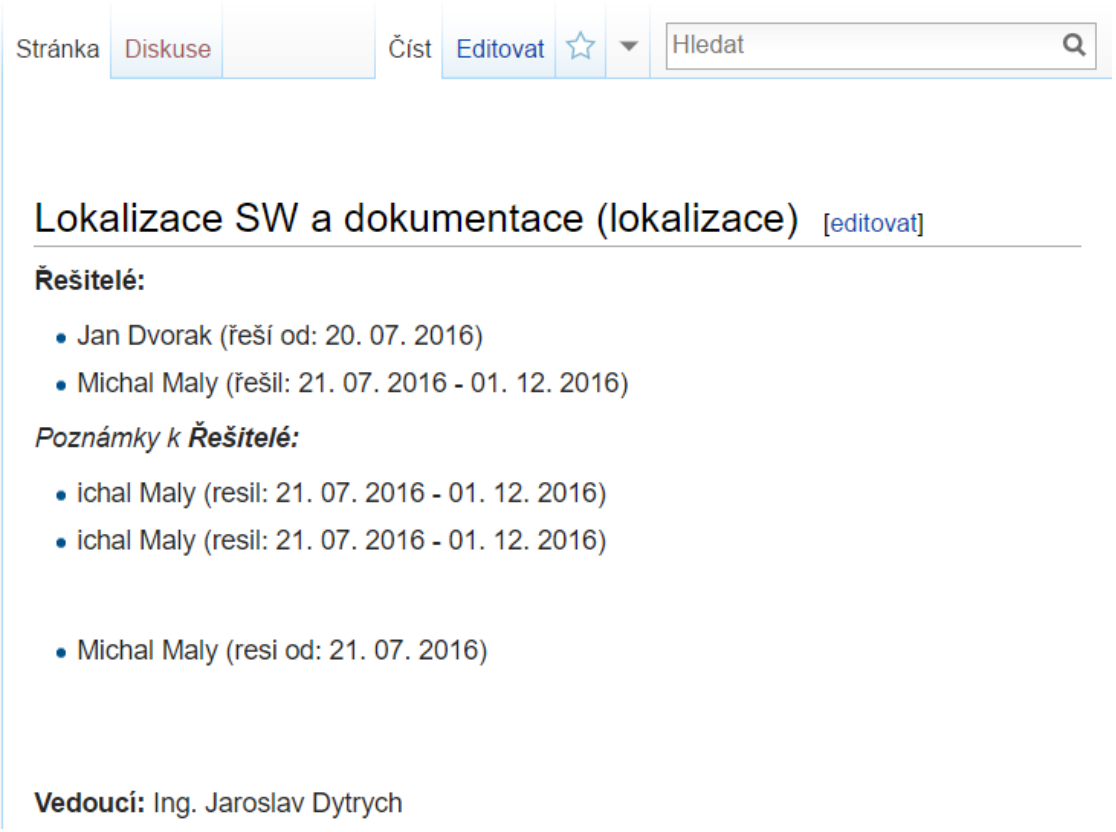
Hlavním problémem současného řešení je veliká chybovost programu, který nebyl schopen ani za optimálních podmínek správně generovat dokumentaci z e-mailů, ani z informačního systému. U generování úkolů a výkazů se k tomu přidal i špatný návrh, který nezahrnoval generování všech potřebných informací a generoval příliš složitou strukturu.

Generování a kontrola skupin projektů a projektů také není bez chyb. Největším problémem je, že program nedokáže rozeznat změnu v databázi od změny provedené ručně uživatelem. Program pozná, že se něco změnilo, a vytvoří poznámku, což někdy může být velmi matoucí a zbytečně zahlcuje hlavičku. Na obrázku 5.1 je vidět ukázka, kdy došlo ke změně v databázi, omylem se změnilo jméno řešitele a program vytvořil zbytečné poznámky, které znepřehledňují celou hlavičku.

I zde se projevuje chybovost programu, kdy po kontrole hlavičky je hlavička zanechána „rozbitá“. Toto například znamená, že projektu může chybět nějaká značka HTML. V tomto textu i v samotném návrhu používám slovo tag, které v nich označuje element pro komentář jazyka HTML, který obsahuje speciální text. Například tag automaticky generované sekce znamená následující komentář HTML v textu:

```
<!-- Začátek automaticky generované sekce -->
```

Tagy nemusí být jenom komentáře HTML, v takovém případě bude tvar tagu explicitně uveden.



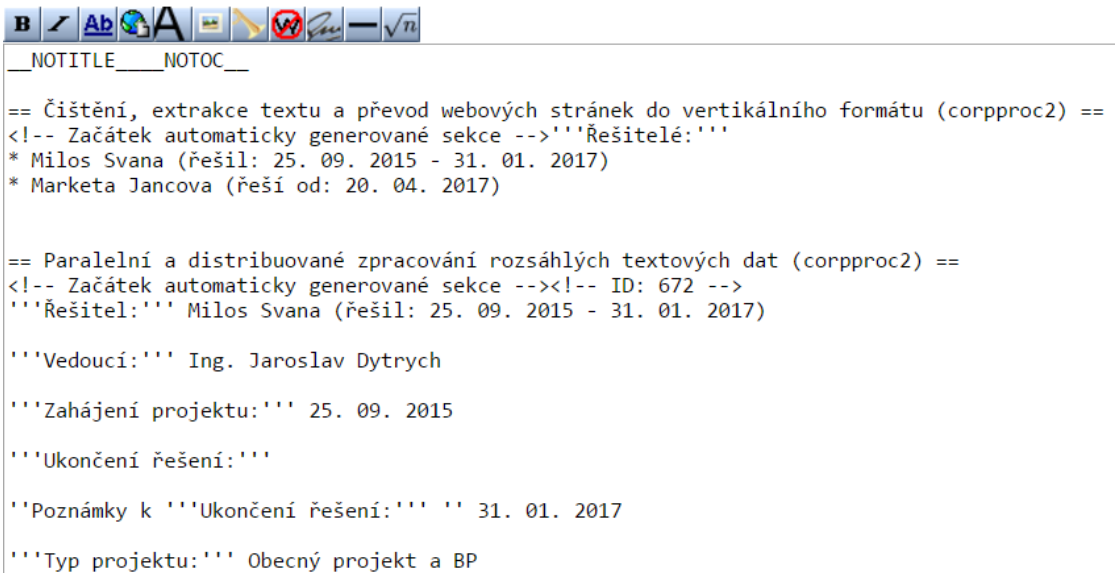
Obrázek 5.1: Ukázka zbytečných poznámek, které tvoří současná verze programu pro automatické generování dokumentace.

V hlavičce také často chybí tag s identifikačním číslem nebo se v hlavičce opakuje tag začátku hlavičky. Toto je k vidění například na obrázku 5.2. Na tomto obrázku je vidět další nedostatek, tentokrát v návrhu, kdy se nadpis projektu generuje před jeho identifikátorem. Toto se mi zdá nelogické a nešťastné řešení.

Program také disponuje malou granularitou nastavení kontroly jednotlivých sekcí. Jde zapnout, nebo vypnout kontrolu pouze celých celků, jako jsou skupiny projektů projekty, úkoly a výkazy. Toto je problém, pokud chceme měnit generování pouze menších celků, třeba vypnout generování 20 projektů, a nechceme ručně psát do WIKI tag negenerovat, což je velmi nepraktické.

Uživatel může na začátek stránky umístit speciální tag `NEGENEROVAT`, který programu řekne, že tuto stránku nemá kontrolovat a má jí vynechat. Tuto proceduru jsem navrhl pro případ, že uživatel nechce, aby program z nějakého důvodu tuto stránku kontroloval a zároveň nemá dostatečná oprávnění, aby to nastavil v informačním systému KNOTIS. Toto se kontroluje na všech zpracovávaných stránkách.

Posledním a podle mého názoru nejhorším nedostatkem programu je jeho implementace. Kód obsahuje funkce, které nemají použitelnou dokumentaci a obsahují stovky



Obrázek 5.2: Ukázka špatného návrhu současného programu na generování dokumentace, kdy se nadpis generuje před identifikátorem projektu. Obrázek také ukazuje chybějící identifikátor u prvního projektu a duplikaci tagu začátku automaticky generované sekce.

řádků někdy i opakujícího se kódu. Na tomto se také podepsal fakt, že tento program se pokoušelo opravit již několik lidí. To znamená, že původní dokumentace k jeho funkčnosti a jednotlivým funkčním blokům je neaktuální a skoro nepoužitelná.

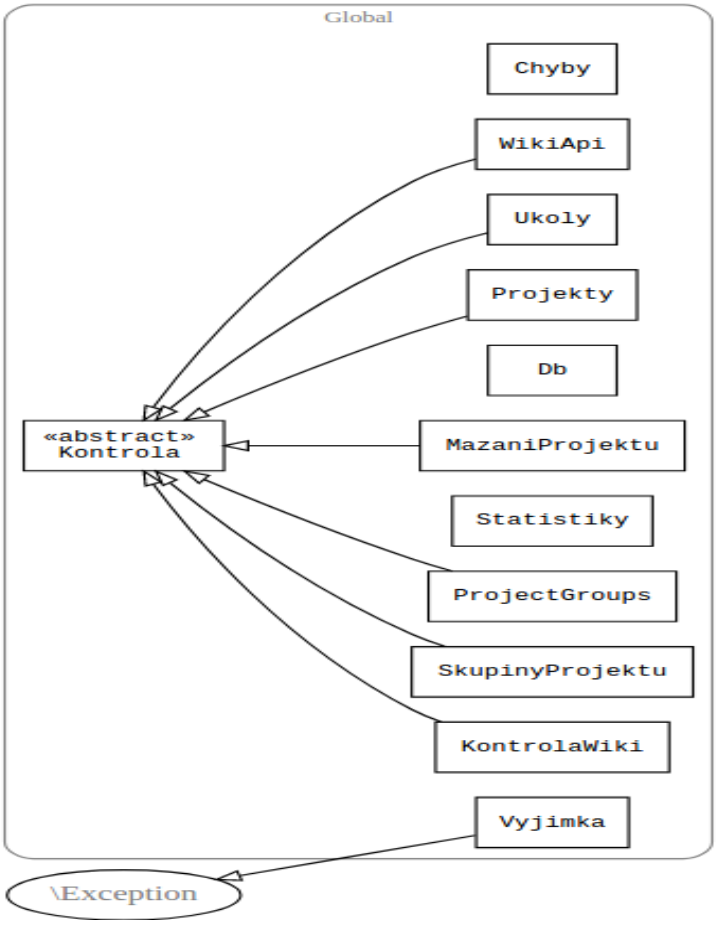
Po nastudování celé problematiky a příslušných prerekvizit jsem se rozhodl, že bude lepší navrhnout a implementovat úplně nový systém generování a kontroly dokumentace. Tento systém využívá, stejně jako jeho předchůdce generování dokumentace pomocí speciálních hlaviček WIKI stránek. Ze starého systému jsem také převzal vzhled hlaviček, generované informace a vzhled tabulku zachycující změny oprávnění prostředků. Tyto věci jsou podle mého názoru dobré a uživatelé jsou na ně zvyklí.

5.2 Návrh nového systému

Před začátkem návrhu jsem si ujasnil jednu základní věc. Není možné v jedné práci vytvořit kvalitní a spolehlivý systém, který bude generovat dokumentaci ze všech dat z KNOTIS a e-mailů, neboť samotná problematika mapování e-mailů ke správným projektům a rozpoznání užitečného obsahu v e-mailu je netriviální. Proto jsem se rozhodl svůj program implementovat jako platformu, která bude poskytovat metody pro komunikaci s Media Wiki API, metody pro statistiku programu, pro zpracování chyb a komunikaci s databází. Do této platformy lze připojit moduly, které slouží pro kontrolu a generování

jednotlivých hlaviček. Například modul projekty slouží pro kontrolu a generování všech projektů z informačního systému KNOTIS.

Kvůli tomuto přístupu jsem musel tuto platformu navrhnout tak, aby měla jasně danou strukturu, bylo možné se v ní snadno orientovat a přidání modulu nebylo složité. Proto jsem systém navrhl a implementoval pomocí objektového návrhu. Díky tomuto přístupu může být každý modul implementován jako třída. Tato struktura je znázorněna v obrázku 5.3. Moduly tedy lze přidávat postupně a v libovolném pořadí.



Obrázek 5.3: Diagram tříd, který ukazuje vazby mezi třídami použitými v novém programu.

Nejdůležitější částí návrhu je postup, jak rozeznat text ručně upravený uživatelem od změny v databázi a neztratit žádný text dopsaný nebo změněný uživatelem. Navrhl jsem univerzální postup, který je možno použít u jakéhokoli řádku v hlavičce. Tento postup využívá hashování obsahu řádku a porovnávání s hashem vytvořeným při minulé kontrole. Také zajišťuje, že program je na rozdíl od jeho předchůdce schopný rozeznat uživatelský text a zachovat se podle toho. Toto je možné vidět v diagramech aktivit, které budu popisovat dále v textu.

Při návrhu tohoto postupu jsem také navrhl vylepšení, jak lze celou kontrolu zrychlit. Opět bylo použito hashování, ale tentokrát za použití dvou hashů. Jeden hash se počítá z WIKI a jeden z databázových dat. Díky porovnání s hashi z minulé kontroly je program schopen přeskočit hlavičky, které neobsahují žádnou změnu, a značně tak zrychlit kontrolu oproti předchozí verzi programu. Tento postup je s různými úpravami použit ve všech kontrolovaných hlavičkách – viz diagramy aktivit, které budou popsány dále v textu.

5.3 Návrh pomocí diagramů aktivit

Po začátku návrhu vyšlo najevo, že tento návrh bude kvůli rozmanitosti použitých dat složitý a obsáhlý. Proto jsem pro vytvoření efektivního řešení použil návrh pomocí diagramů aktivit (dále jen diagramů). Hlavní přínosy tohoto přístupu jsou, že je snadno pochopitelný, velice názorný a pokud je podle něj implementován program, tak tyto diagramy slouží jako velice dobrá dokumentace [7].

Všechny diagramy jsou umístěny jako přílohy k této práci a rozděleny do sekcí. Sekce se nazývají podle modulu, který zobrazují. Pod nadpisem každé sekce budou nadpisy všech diagramů, které patří do dané sekce. Každý diagram obsahuje na svém začátku komentář se svým názvem a krátkým popisem. V textu se budu vždy odkazovat na nadpis celého modulu.

5.4 Celkový průběh

Diagram Celkový průběh A.1 ukazuje obecnou strukturu průběhu celého programu, stejně jako zpracování a uložení statistik a chyb. První část diagramu popisuje nezbytné kroky, které jsou potřeba pro běh programu, jako je přihlášení do Wiki, databáze, načtení nastavení a kontrolu úspěšného provedení všech těchto akcí. V levé dolní části jsou zobrazeny jednotlivé moduly programu. V pravé dolní části se nachází zpracování statistik, chyb a jejich logování.

5.5 Kontrola začátku stránky

Diagram Kontrola začátku stránky A.2 znázorňuje návrh kontroly začátku stránky. Na úplném začátku stránky se musejí nacházet speciální tagy `__NOTOC__NOTITLE__`, které WIKI řeknou, že nemá generovat tabulku obsahu stránky a nadpis stránky. Toto zajišťuje, že na začátku stránky je naše speciální hlavička s námi vygenerovaným nadpisem a ne automaticky vygenerovaným WIKI nadpisem, který se bere z adresy stránky. Poté se kontrolují tagy automaticky generované sekce, které ohraničují hlavičku. Mezi speciálními tagy a hlavičkou může být uživatelský text. Toto jsem do návrhu zakomponoval proto, kdyby uživatel chtěl na úplný začátek dopsat nějaké důležité informace.

5.6 Kontrola nadpisu

Diagram Kontrola nadpisu A.2 znázorňuje kontrolu nadpisu hlavičky. Navrhl jsem postup, který nejprve zkontroluje, jestli má nadpis správné tagy. V případě nadpisu stránky nebo modulu `== Nadpis ==`. Poté zkontroluje text nadpisu. Pokud tento postup zjistí, že se text nadpisu liší a byl upraven uživatelem, tak vytvoří o této události poznámku s textem změněného nadpisu a nadpis stránky opraví.

5.7 Kontrola struktury tagu

Diagram Kontrola struktury tagu A.2 znázorňuje kontrolu struktury tagu. Tento modul je navržený tak, že kontroluje, jestli se text nachází mezi správnými tagy. Pokud jsou tagy jen poškozené, tak je tento modul umí opravit.

5.8 Kontrola výskytu

Diagram Kontrola výskytu A.2 znázorňuje kontrolu výskytu určitých dat v datech hlavičky. Tento modul zabraňuje znovu vytvoření a následné duplikaci dat, kdy se nějakým způsobem, třeba zásahem uživatele, dostane před hledaná data v hlavičce jiný text. Pokud se hledaná data nenacházejí v hlavičce, modul vrátí nenalezeno. Jinak vrátí nalezená data a data, která se nacházela před nimi. Pokud se ovšem před hledanými daty nachází klíčové slovo, pak modul tuto hlavičku identifikuje jako poškozenou, vrací stav nenalezeno pro modul, který jej volal, a zaznamená varování. Klíčové slovo je speciální slovo, které identifikuje řádek s daty a říká nám, jaká data máme na řádku očekávat. Vždy máme v hlavičce jen jedno klíčové slovo jednoho druhu, například *Vedoucí, Řešitel,* atd., jak můžeme vidět na obrázku 6.7.

5.9 Seznam skupin projektů

Diagram Project Groups A.3 znázorňuje modul pro generování a kontrolu stránky se seznamem skupin projektů (Project Groups). Jedná se o speciální stránku, která zobrazuje všechny skupiny projektů z KNOTIS. Tato stránka byla v minulosti vytvářena ručně. To nebylo praktické, a proto jsem navrhl modul, který toto bude dělat automaticky.

Tento modul je znázorněn na diagramu Project Groups A.3. První část diagramu obsahuje načtení informací z databáze a zjištění, jestli stránka na dané adrese již existuje. Pokud neexistuje, tak bude vytvořena. Návrh vytvoření stránky je vidět na diagramu A.3 Vytvoření stránky Project Groups.

Pokud stránka existuje, tak se zkontroluje její obsah. Tento postup je vidět v levé dolní části diagramu pro kontrolu modulu Project Groups. Načtou se data stránky a kontroluje se tag negenerovat.

V tomto modulu jsem nepoužil hashování celé hlavičky, protože počet skupin projektů se pohybuje do desítky a nijak výrazně by to nezrychlilo kontrolu této jedné stránky,

naopak by to zbytečně zkomplikovalo návrh. Modul naopak podporuje možnost změny adresy a nadpisu stránky z informačního systému KNOTIS.

5.10 Skupiny projektů

Diagram Skupiny projektů A.4 znázorňuje modul pro kontrolu skupin projektů. Skupin je v KNOTIS uloženo do desítky a v dlouhodobém horizontu se neplánuje nějaké drastické navýšení. Oproti starému programu jsem do návrhu zakomponoval možnost, že na jedné stránce může být hlavička skupiny projektu, ale i hlavička projektu. Díky této funkci jsem nemohl vypočítat hash celé automaticky generované sekce a kontrolovat jenom jeho změnu. Musely by se načítat data pro skupiny a projekty pro každou URL zvlášť, což by bylo výpočetně náročné. Místo toho se načtou data skupin nebo projektů a seriově se kontrolují nejprve hlavičky skupin, a pak až hlavičky projektů. Toto také umožňuje řadit hlavičky. Nejprve budou uvedeny hlavičky skupin a pak hlavičky projektů.

Doposud se v tomto dokumentu používal pojem hlavička pro popis speciální hlavičky na začátku WIKI stránky, protože to bylo lépe pochopitelné. V kontrole a generování hlaviček se ale často setkáváme s tím, že se ve WIKI hlavičce vyskytuje více hlaviček. Kvůli lepší orientaci se dále v textu bude označovat speciální hlavička na začátku WIKI stránky automaticky generovanou sekcí a hlavičkou bude míněna hlavička projektu, skupiny projektů, atd, která je umístěna v této automaticky generované sekci.

Horní část diagramu popisuje načítání dat skupin projektů. Nejprve modul načte základní data, jako je *ID, URL, název* a všechny hashe skupin. Poté skupiny seskupí podle URL, zkontroluje, jestli se URL nevyskytuje ve WIKI výjimkách, a kontroluje postupně všechny URL. Pokud není adresa ve výjimce, načtou se podrobná data všech skupin, které mají být na tomto URL přítomny. Tento způsob načítání dat jsem navrhl pro optimalizaci výkonu, kdy se snažím o rovnováhu objemu načtených dat a počtu přístupů do databáze, které jsou časově náročné. Poté následuje kontrola existence stránky.

Pokud stránka neexistuje, tak je vytvořena, jak ukazuje diagram vytvoření stránky skupiny projektů A.4. Tento diagram by se dal rozdělit na dvě sekce. Pravou pro případ, kdy na tomto URL máme vytvořit jednu skupinu. A levou pro případ, kdy máme na tomto URL vytvořit více skupin. Všechny moduly jsou navrženy tak, že evidují statistiky práce s každou samostatnou hlavičkou, tzn. dobu od začátku zpracování po konec zpracování hlavičky.

Pokud stránka existuje, tak se zkontroluje její obsah podle dolní části diagramu skupina projektů A.4. Modul načte všechny hlavičky v automaticky generované sekci a zkontroluje hlavičky skupin projektů. Zde jsem pro urychlení kontroly navrhl postup, kdy se vypočítá hash celé načtené hlavičky a druhý hash, který se vypočítá pouze z dat načtených z databáze, a poté se tyto hashe porovnají s hashi uloženými z minulé kontroly programu. Pokud se rovnají, tak modul ví, že se hlavička nezměnila a může ji přeskočit. Jelikož se hlavičky skupin často nemění, tak tento postup zrychlí celý modul. Pokud se ale hashe nerovnají, tak modul hlavičku zkontroluje. Kontroly hlaviček modulů jsou pro lepší orientaci a přehlednost rozděleny do subdiagramů. Jako první se kontroluje řádek s nadpisem skupiny. Tento postup je zobrazen diagramem kontrola nadpisu A.2 a byl

popsán v sekci Kontrola nadpisu. Další kroky kontroly hlavičky budou popsány dále v textu.

Po těchto kontrolách se sestaví automaticky generovaná sekce, kde se nejprve vloží hlavičky skupin projektů seřazené podle abecedy. Pokud jsou přítomny, tak se vloží seřazené hlavičky projektů. Přidá se text, který má být přemístěn z hlavičky pod automaticky generovanou sekci a na závěr se přidá zbytek stránky, který byl umístěn pod touto automaticky generovanou sekcí před kontrolou. Až modul zkontroluje všechny URL uložené v databázi, tak se ukončí a program spustí další modul, pokud je přítomen. Tento postup je vidět v pravé části diagramu.

Kontrola vedoucího

Dále se kontroluje vedoucí skupiny projektů. Tento modul je zobrazen diagramem Kontrola vedoucího A.4 a používá se i pro kontrolu vedoucího projektu. Modul porovná údaje vedoucího, načtené z WIKI, s daty načtenými z databáze. Pokud se nerovnají, tak se nejprve podívá, v čem se liší. Pokud se liší pouze v titulu, titul je opraven na hodnotu z databáze. Tato vlastnost modulu byla navržena z důvodu, že změny titulu se budou u vedoucích provádět nejčastěji. Pokud se liší v něčem jiném, tak modul vypočítá hashe a zjistí, jestli uživatel něco nezměnil ručně. Pokud ano, tak vytvoří poznámku, že uživatel změnil vedoucího. Vždy budeme mít pouze jednu tuto poznámku. Nemůže se tedy stát, že se pod vedoucím bude vyskytovat více těchto poznámek za sebou. Toto je z důvodu, že když se provede změna pouze v databázi, tak modul vytvoří jinou poznámku, a to poznámku *Předchozí vedoucí*. Program nově obsahuje postup pro vedení historie vedoucích. Pokud již vedoucí je v předchozích vedoucích, tak se zkontroluje aktuálnost dat v poznámce, jako je titul, jméno, příjmení a identifikátor. Ostatní vedoucí v poznámkách se nekontrolují, protože se tato historie uchovává pouze ve WIKI a ne v databázi. Pokud bychom chtěli kontrolovat i jejich data, tak by je modul musel najít a načíst z databáze, což by bylo velice výpočetně náročné a běh programu by se značně prodloužil. Po všech těchto úkonech modul opraví data vedoucího.

Kontrola Redmine

Redmine je volně dostupný nástroj pro správu úkolů a sledování chyb projektů. Výzkumná skupina KNOT má tento nástroj nainstalovaný na svém serveru, protože některé skupiny projektů a projekty tento nástroj používají. Jeho využití je výhodné zejména u velkých projektů s mnoha úkoly, jejichž správa by v KNOTIS nebyla přehledná. V databázi je uložena URL, která slouží pro přístup do daného projektu. Starý program nepodporovat generování a kontrolu této informace v hlavičce.

Diagram Kontrola Redmine A.4 zobrazuje modul pro kontrolu adresy do Redmine. Tento modul se využívá v kontrole skupiny i projektu. Jak bylo řečeno výše, tak tento nástroj používají jen některé skupiny a projekty. Proto jsem navrhl postup, který když není odkaz na Redmine v databázi, tak ho v hlavičce nevytvoří. Takto jsem se vyhnul zbytečnému řádku v hlavičce, který by byl u většiny hlaviček prázdný.

Problematickým bodem tohoto postupu je kontrola hlavičky, ve které se nachází odkaz do Redmine, který ale již není v databázi. Pak může být řádek identifikován jako text navíc a přesunut pod automatiky generovanou sekci. Pro tento účel jsem vytvořil další postup, který se nachází v pravé části diagramu.

Nejprve se modul podívá, jestli je uložen hash z minulé kontroly. Pokud není, tak vytvoří poznámku *Uživatel přidal Redmine ručně*. Pokud je uložen hash poslední kontroly, tak vypočítá hash z dat a porovná ho s hashem poslední kontroly. Pokud se nerovnají, tak se vytvoří poznámka *Uživatel upravil Redmine ručně*. Tento postup také počítá i s variantou, že se může odkaz na Redmine opět do databáze přidat.

Kontrola Externí URL

Některé projekty v informačním systému KNOTIS mají svoji vlastní webovou stránku, která leží mimo KNOTIS a KNOT WIKI. Doposud tento údaj nebyl nikde systematicky uložen. Proto jsem vytvořil v databázi sloupce pro tyto adresy a vytvořil modul, který je bude generovat a kontrolovat v hlavičkách skupin projektů a projektů ve WIKI.

Tento modul je popsán diagramem Kontrola externí URL A.4 a využívá se v kontrole skupin projektů a projektů. Skupina projektů může mít také externí URL.

Postup kontroly je stejný, jako u kontroly Redmine.

Kontrola Prostředků

Skupiny projektů a projekty mohou mít přidělené prostředky na serverech výzkumné skupiny KNOT. Ve WIKI hlavičce se vytváří speciální tabulka, do které jsou zaznamenány aktuální stavy prostředků. Jako je název prostředku, cesta k prostředku, jaké oprávnění mají skupiny nebo projekt k danému prostředku, ke kterému projektu daný prostředek patří a název serveru, kde je prostředek umístěn.

Kolonka pro oprávnění je speciální, obsahuje totiž historii stavů oprávnění. Oprávnění může nabývat třech stavů: *žádné, čtení, zápis*. Oprávnění *žádné znamená*, že projekt nebo skupina měly v minulosti přístup k tomuto prostředku, ale tento přístup jim byl odebrán. Toto má ale nevýhodu v tom, že nevíme, jestli měl projekt v minulosti právo zápisu nebo čtení. Pro tento účel jsem převzal postup, který využívá starý program pro generování dokumentace [11]. Pokud měl někdy projekt právo zápisu, tak po změně oprávnění se k názvu oprávnění přidá „ * “, jak je vidět v tabulce 5.1, nebo v diagramu Kontrola prostředků A.4, který zobrazuje modul pro kontrolu a generování tabulky prostředků. Tento modul je využíván v modulech pro kontrolu skupiny projektů a projektů. Horní část diagramu znázorňuje postup pro zjištění, jestli se v hlavičce tabulka nachází a jestli její záhlaví má správný formát. Prostřední část diagramu znázorňuje postup načtení dat tabulky a vypočítání hashů. Modul vypočítá hash dat načtených z WIKI a hash dat načtených z databáze. Porovná tyto dva hashe a pokud se oba rovnají, tak přeskakuje tabulku, protože se v ní nic nezměnilo. Pokud se ale hashe nerovnají, kontroluje jednotlivé řádky tabulky. Toto je znázorněno v dolní části diagramu a používají se l tomu postupy popsané výše v textu.

Výchozí oprávnění	Požadované oprávnění	Výsledné oprávnění
Žádné	Zápis	Zápis
Žádné*	Zápis	Zápis
Žádné	Čtení	Čtení
Žádné*	Čtení	Čtení*
Čtení	Zápis	Zápis
Čtení*	Zápis	Zápis
Čtení	Žádné	Žádné
Čtení*	Žádné	Žádné*
Zápis	Čtení	Čtení*
Zápis	Žádné	Žádné*

Tabulka 5.1: Tabulka pro změny oprávnění v tabulce prostředků. Tento postup byl převzat z diplomové práce pana Aleše Vavřínka [11].

Kontrola projektů Skupiny projektů

Diagram Kontrola projektů skupiny projektů A.4 zobrazuje modul pro kontrolu odkazů na projekty, které patří pod skupinu projektů. Vrchní část diagramu zobrazuje postup, který nejprve nečte všechny odkazy na projekty a zjistí, jestli se změnila nějaká data. Je použit postup s vypočítáním hashů dat z WIKI a databáze, jako u kontroly prostředků. Pokud se nic nezměnilo, modul odkazy na projekty přeskočí. Jelikož skupiny mohou obsahovat i stovky projektů, tak tento způsob celou kontrolu hlavičky značně zrychluje. Pokud se něco změnilo, tak modul přistupuje ke kontrole každého řádku s odkazem. Toto je zobrazeno v levé dolní části diagramu.

Mnou navržený postup umí rozeznat, jestli je odkaz vnitřní, tedy vede na nějakou WIKI stránku v KNOT WIKI, nebo vnější a vede na nějakou jinou URL. Media Wiki tyto odkazy umí barevně odlišit, takže uživatel na první pohled vidí, jestli je to WIKI stránka nebo nějaký externí odkaz. Nově navržený postup také umí rozeznat a opravit odkazy, které mají rozbitou strukturu. Odkaz s rozbitou strukturou bude přemístěn na konec do sekce nazvané *Nekonzistentní projekty* a na jeho místo bude vygenerován nový odkaz se správnou strukturou. Díky tomuto neztratíme žádná uživatelská data a opravíme chybné odkazy. Uživatel WIKI pak bude moci sám rozhodnout, jestli tyto nekonzistentní odkazy chce v hlavičce nechat, nebo je vymaže. Další novinkou je, že pokud byl projekt již vymazán z databáze, tak modul tento odkaz přesune do sekce pro *Projekty navíc*. Díky tomuto se na stránce skupiny projektů uchovají i staré projekty. Samozřejmostí je oprava neaktuálních dat v odkazu. Zde se nekontroluje uživatelský obsah. Toto řešení jsem zvolil z důvodu, že stránky skupin projektů působí jako takový

rozcestník s ukazateli na stránky jednotlivých projektů, které jsou uloženy v informačním systému KNOTIS, a nechtěl jsem do této sekce, kde již teď je velmi mnoho odkazů, vkládat další zbytečný obsah. Pokud by ale uživatel chtěl umístit nějakou svou důležitou poznámku, má možnost ji umístit pod odkaz na projekt na dalším řádku. Kontrola tuto poznámku bude ignorovat a přidá ji zpátky pod projekt. Toto řešení považuji za dobrý kompromis mezi přehledností této sekce a možností uživatelského textu.

Po skončení kontroly všech odkazů budou odkazy nejprve seřazeny podle jejich stavu, kdy pořadí je následující: *Řešený, Ukončený, K rozhodnutí, Nezadaný*. Poté budou seřazeny podle abecedy. Toto řazení probíhá pro každý stav samostatně. Poté se přidají projekty navíc a projekty s nekonzistentní strukturou, jak je vidět v pravé dolní části diagramu. Pokud máme nějaký nekonzistentní projekt, tak modul vytvoří varování.

5.11 Projekty

Diagram Projekty A.5 znázorňuje modul pro kontrolu projektů. Projektů jsou v informačním systému KNOTIS uloženy stovky a toto číslo každoročně narůstá, jak bylo vidět na obrázku 4.1. Jako v případě modulu pro kontrolu skupin i tento modul počítá s možností hlaviček skupin a projektů na jedné stránce. Kvůli tomuto se nepočítá hash celé automaticky generované sekce, jak bylo popsáno v kontrole skupiny projektů.

Jak bylo popsáno výše, kvůli velkému množství dat jsem navrhl postup, který načítá projekty a jejich data postupně. Tento postup je znázorněn v horní části diagramu. Nejprve se načte identifikátor a URL všech projektů z databáze, které mají zapnuté generování hlavičky projektu. Poté se projekty seskupí podle URL a vyjmou se URL, které jsou umístěny ve WIKI výjimkách. WIKI výjimky a jejich funkce byly popsány v kapitole Data 4. Následně se vezme N URL a pro ty se načtou podrobná data všech projektů, které jsou na těchto URL přítomny. Díky tomuto postupu vždy pracujeme pouze s N projekty, což snižuje paměťovou náročnost programu a zlepšuje výkon programu. Nevýhodou je, že musíme přistupovat vícekrát do databáze. Proto je nutné najít N takové, aby bylo co nejméně přístupů do databáze a modul nepracoval s velkým množstvím dat. Hledání N bude více rozebráno v sekci Testování 7. Po načtení podrobných dat modul sekvenčně kontroluje jedno URL po druhém.

když stránka na daném URL neexistuje, modul ji vytvoří. Toto je znázorněno diagramem Vytvoření stránky projektu A.5. Postup vytvoření stránky závisí na počtu hlaviček projektů, které se mají vytvořit. Situaci, kdy se má vytvořit pouze jedna hlavička, zobrazuje pravá část diagramu. Nejprve modul zjistí, jaké má projekt v informačním systému KNOTIS nastaveno generování. Ve staré verzi programu šlo nastavit, pouze to zde se má generovat hlavička se všemi informacemi, nebo jenom hlavička s nadpisem. Nyní návrh počítá s pěti úrovněmi generování:

- *Negenerovat nic*
- *Generovat nadpis*
- *Generovat Hlavičku + Nadpis*

- *Generovat Hlavičku + Nadpis + Úkoly*
- *Generovat Hlavičku + Nadpis + Úkoly + Výkazy.*

Tato informace je uložena v databázi. Úkoly je modul programu, který bude popsán dále v textu. Návrh počítá s možností přidání dalších modulů a možností generování.

Pokud má hlavička nastaveno generovat jenom nadpis, tak se vygeneruje hlavička s identifikátorem a nadpisem. Pokud se má generovat celá hlavička, tak se k nadpisu přidají další údaje, jako je třeba vedoucí projektu, řešitelé projektu (projekt může mít 0..N řešitelů), datum začátku a ukončení řešení projektu, typ projektu, odkaz na Redmine (pokud ho má projekt nastaven), externí URL (pokud existuje), tabulku prostředků a pokud máme generovat i úkoly nebo výkazy, tak odkazy na jejich stránky. Pokud máme na WIKI stránce generovat více než jednu hlavičku, modul nejprve seřadí hlavičky podle názvu a zkratky a pak je jednu po druhé vygeneruje podle postupu pro generování jedné hlavičky, který byl popsán výše.

Když stránka na daném URL existuje, modul zkontroluje tag negenerovat a pokud není přítomen, začne s kontrolou podle dolní části diagramu Projekt. Kontrola využívá submoduly, které budou popsány dále v textu. Po dokončení kontroly hlavičky projektu modul seřadí hlavičky podle názvu a zkratky. Pokud jsou přítomny hlavičky skupin, tak je přesune před hlavičky projektů, přidá zbytek stránky a jde na další URL jako u kontroly skupin projektů. Až dojdeme na konec N seskupených URL, načteme další. Pokud již další URL nemáme, modul se ukončí.

Prvním krokem v kontrole hlavičky projektu je vypočítání hashů hlavičky. Pokud se rovnají, hlavička se přeskočí, pokud ne, hlavička se zkontroluje. Tento postup byl již popsán v kontrole hlavičky skupiny projektů.

Pokud modul kontroluje hlavičku, nejprve proběhne kontrola nadpisu projektu. Pro tento účel se používá modul pro kontrolu nadpisu popsaný výše v textu. Dále proběhne kontrola vedoucího, která již také byla popsána dříve v textu.

Následuje kontrola řešitele projektu. Tento modul je zobrazen diagramem Kontrola řešitele A.5. Projekt může mít 0..N řešitelů. Menší komplikace nastává, pokud se mění počet řešitelů z více na méně. Pro všechny možné případy diagram obsahuje větve s postupy, jak se s touto komplikací vypořádat. Všechny tyto postupy opět využívají hashe pro zjištění uživatelské změny. Pokud uživatel něco ručně upravil, vytvoří se poznámka, která bude umístěna pod celým blokem s řešiteli místo pod každým upraveným řešitelem zvlášť z důvodu větší přehlednosti.

Po řešitelích se kontroluje řešení projektu. Toto znázorňuje diagram Řešení projektu A.5. Tento modul zajišťuje kontrolu řádků se zahájením a ukončením řešení projektu. Jako datum začátku řešení projektu je zvoleno nejmenší datum zahájení řešení ze všech řešitelů projektu. Pokud projekt nemá řešitele, jako datum zahájení řešení se vezme datum vložení projektu do informačního systému KNOTIS. Jako datum ukončení se vezme naopak největší datum ukončení řešení projektu ze všech řešitelů. Pokud má nějaký řešitel nastaveno ukončení řešení na nekonečno nebo projekt nemá žádného řešitele, tak se místo data nastaví pomlčka.

Dalším řádkem, který se kontroluje, je typ projektu. Tento modul zobrazuje diagram Kontrola typu projektu A.5. Projekt může mít různé typy, jako například *Obecný projekt,*

Obecný projekt + Bakalářská práce, atd. Menší komplikací je, že v databázi jsou uloženy pouze čísla stavů a názvy jsou uloženy v nastavení informačního systému KNOTIS. Tento modul používá postup porovnání hodnoty z minulé kontroly a při uživatelské změně vytvoření poznámky jako předchozí moduly, ale nepočítá a neukládá hash. Místo toho ukládá název typu projektu. Tento způsob jsem zvolil kvůli optimalizaci, kdy hash má pevnou délku 16 znaků, přičemž i nejdelší název typu této délky nedosahuje.

Dále se v hlavičce kontrolují skupiny projektů, do kterých projekt patří. Tento modul je zobrazen diagramem Kontrola skupin projektů A.5. Skupiny projektů jsou reprezentovány odkazy. Modul obsahuje postup pro zjištění uživatelské změny pomocí hashe a vytvoření případné poznámky. Pokud projekt nepatří do žádné skupiny, tak se místo odkazů vygeneruje pomlčka.

Po kontrole skupin projektů, ve kterých je daný projekt, se kontroluje řádek s odkazem na Redmine a řádek s odkazem na externí URL. Oba tyto moduly a diagramy již byly popsány dříve v textu A.4. Následuje kontrola prostředků, která již byla také popsána výše.

Poslední moduly, které kontrolují hlavičku projektu jsou moduly kontroly odkazu na úkoly a kontroly odkazu na výkazy. Tyto moduly popisují diagramy Úkoly řešitele a Výkazy k projektům A.5. Tyto moduly mají stejnou logiku struktury, proto je budu popisovat jako jeden celek.

Modul nejprve zjistí, jestli má projekt zapnuté generování tohoto řádku. Pokud nemá a řádek je v hlavičce přítomen, tak na tento řádek přidá slovo *Nekontrolováno*, čímž dá uživateli najevo, že tento odkaz není kontrolován a může být neaktuální. Díky tomuto postupu jsou již vygenerované úkoly vždy dostupné a nestávají se z nich sirotčí stránky. Pokud je zapnuté generování a informace na řádku nejsou aktuální, tak se vypočítá hash, a pokud tuto neaktuálnost vytvořil uživatel, vytvoří se poznámka. Generovaný odkaz vede na speciální stránku. Podrobnosti o této stránce určují moduly pro její kontrolu.

Platforma

Moduly pro kontrolu skupin projektů a projektů dohromady s moduly pro chyby, statistiku a ostatními pomocnými moduly, které budou popsány v sekci 6, tvoří platformu pro automatické generování dokumentace z informačního KNOTIS. Všechny ostatní moduly, které budou popsány dále v návrhu, se dají popsat jako rozšiřující zásuvné moduly. To proto, že bez těchto modulů může platforma autonomně fungovat a dokumentace bude obsahovat všechny důležité informace a funkčnosti. Tyto zásuvné moduly byly navrženy jako rozšíření této práce, aby demonstrovaly možnosti rozšiřitelnosti platformy a její potenciál.

5.12 Úkoly

Zásuvný modul úkoly je zobrazen na diagramu Úkoly A.6. Pro každý projekt, který má alespoň jednoho řešitele, je vygenerována stránka, která bude obsahovat všechny úkoly, stavy úkolů a připadané komentáře ke stavům všech řešitelů projektu. Horní část diagramu znázorňuje načtení úkolů a jejich podrobných dat. Nejprve načteme identifikátory

řešitelů, kteří nemají zakázané generování úkolů, a projekty, které mají zapnuté generování úkolů. Nově jsem navrhl postup, kdy vedoucí může zakázat generování úkolů nebo výkazů řešiteli projektu. Tento zákaz se vztahuje pouze na daného řešitele v daném projektu. Tato vlastnost se může hodit v případě, že řešitel pracuje na úkolech v projektu, které nejsou určeny k vidění pro ostatní uživatele WIKI, ale jenom pro vedoucí, kteří si je mohou prohlédnout v informačním systému KNOTIS. Po vybrání všech projektů a řešitelů, kterým mají být generovány úkoly, tyto řešitele seskupíme podle identifikátoru projektu. Poté vybereme N seskupených identifikátorů projektu a načteme pro ně podrobná data. Postup je stejný jako u načítání projektů. Tento postup byl popsán již dříve v textu. Jednotlivé URL úkolů projektů se liší v identifikátoru projektu, který je součástí URL. Tento postup jsem zvolil kvůli jednoznačné identifikaci stránky, jelikož identifikátor musí vždy být jednoznačný.

Pokud stránka neexistuje, bude vytvořena, jak je zobrazeno na diagramu Vytvoření stránky úkolu A.6. Na začátku stránky vytvoří modul nadpis s názvem projektu a jeho identifikátorem. Poté jsou vytvořeny hlavičky všech řešitelů. V těchto hlavičkách jsou dále vytvořeny všechny úkoly, které řešitel má, a k těmto úkolům jsou také vytvořeny všechny změny jejich stavů. Pokud nějaký stav obsahoval komentář zadaný do informačního systému KNOTIS, tak tento bude vytvořen pod daným stavem. Popsaný postup jsem zvolil pro rovnováhu informačního sdělení hlavičky a její velikosti. V informačním systému se totiž vede celé historie úkolů i jejich stavů s různými informacemi. Tento seznam je ale velice nepřehledný a často matoucí.

Pokud stránka existuje, bude se kontrolovat její obsah. Toto znázorňuje levá část diagramu Úkoly A.6. Nejprve se pomocí modulu Kontrola nadpisu kontroluje nadpis stránky. Tento modul byl již popsán dříve. Poté se kontrolují jednotlivé hlavičky. Nejprve se kontroluje, jestli hlavička neobsahuje tag `(!NEAKTUALIZOVÁNO!)` .

Řešitelé projektu se mohou často měnit. Někdo nový přijde nebo někdo projekt opustí. Otázkou bylo, co dělat s jejich úkoly. Proto jsem vytvořil postup, kdy pokud řešitel již není v databázi, tak modul zkontroluje hlavičku, jestli se v ní nevyskytuje nějaký uživatelský obsah, který uživatel ručně upravil nebo přidal. Pokud neexistuje, tak je hlavička řešitele smazána. Pokud se ale tento uživatelský obsah vyskytuje, tak je do hlavičky přidán tag neaktualizováno. Pokud modul kontroly úkolu narazí na začátku hlavičky na tento tag, tak hlavičku přeskakuje. Díky tomuto postupu se žádná uživatelská data neztratí. A naopak data, která již nejsou potřeba, se vymažou.

Pokud se tedy tag neaktualizováno nevyskytuje, modul zkontroluje nadpis hlavičky. Pokud v něm uživatel něco ručně opravil, vytvoří se poznámka. Tento postup využívá hashe, stejně jak bylo popsáno v ostatních modulech. Poté se postupně načítají úkoly řešitele. Vypočítají a porovnají se hashe a pokud se rovnají, tak je úkol přeskočen. Pokud ne, tak se zkontroluje.

Jako první se v úkolu kontroluje řádek s datem do kterého je nutné daný úkol vypracovat. Toto znázorňuje diagram Kontrola vypracovat do A.6, kde pokud uživatel opět něco ručně změnil, tak se vytvoří poznámka. Toto platí i o kontrole stavu úkolu. Pro kontrolu komentáře ke stavu jsem navrhl trochu jiný postup. A to takový, že pokud tento komentář byl upraven uživatelem, modul ho změní na poznámku, kterou umístí pod komentář, který byl doplněn z databáze. Úkoly jsou v hlavičce řešitele seřazeny podle

stavu úkolu. Stav může být *Zadaný, Řešený, Smazaný, Nezadaný, Vyřešený, Akceptovaný, Vrácený, Nejasný, Zodpovězený*. A jednotlivé změny stavů úkolu jsou řazeny podle procent splnění a poté podle data, kdy nahoře jsou vždy ty nejnovější změny stavů.

Po skončení kontroly jsou hlavičky řešitelů seřazeny podle jmen řešitelů a stránka je sestavena dohromady a uložena.

5.13 Mazání projektů

Pokud administrátor vymaže projekt z informačního systému KNOTIS, tak nastává otázka, jestli nechat jeho WIKI stránku na WIKI, nebo ji smazat. Proto jsem přišel s postupem, který se snaží rozlišit obsah stránky a podle toho rozhodne, jestli stránku smaže, vymaže pouze automaticky generovanou sekci nebo pouze hlavičku projektu, nebo nechá stránku nezměněnou. Tento postup využívá modul pro mazání projektů, který je znázorněn diagramem Mazání projektů A.7.

Toto je jediný modul, který je volán z informačního systému KNOTIS a není spouštěn v intervalech. Spustí se ve chvíli, kdy uživatel chce vymazat projekt z KNOTIS. Modul zkontroluje WIKI stránku projektu a výsledek vrátí systému KNOTIS.

Modul nejprve zkontroluje, jestli stránka existuje. Pokud ano, tak zjistí, jestli jsou na stránce nějaká data. Pokud ne, tak ji vymaže. Pokud ano, tak vyjme automaticky generovanou sekci a zjistí, jestli se v ní nachází hlavička mazaného projektu. Pokud ne, tak se podívá, jestli se v ní nacházejí nějaké další hlavičky. Pokud v ní nejsou žádné další hlavičky, tak vymaže automaticky generovanou sekci. Pokud se v automaticky generované sekci nachází hlavička mazaného projektu, tak modul zkontroluje, jestli se v ní nachází uživatelský text a pokud ano, tak celou stránku nechá v původní podobě. Pokud se uživatelský text nenachází, tak vymaže hlavičku a zkoumá, jestli se v automaticky generované sekci vyskytují další hlavičky a pokud ne, tak ji smaže. Pokud smazal automaticky generovanou sekci, podívá se, jestli se ve zbytku stránky nachází nějaký text. Pokud ne, stránka je smazána, a pokud ano, tak uloží upravenou stránku zbavenou zbytečných dat. A na WIKI nebudou v éteru nepotřebné stránky.

Kapitola 6

Implementace

Jak bylo zmíněno v kapitole 3.6, plánuje se aktualizace na novou verzi systému Ubuntu. Proto jsem před samotnou implementací programu převedl informační systém KNOTIS do PHP verze 7 a otestoval aktualizaci KNOT WIKI na novou verzi 1.28.1. V KNOTIS bylo potřeba nahradit funkce balíku `MySQL.md` za funkce z balíku `MySQLi`, jelikož tento balík není v nové verzi podporován. Také bylo potřeba upravit některé funkce pro regulární výrazy, které využívaly přepínač `G`, jelikož tento byl rovněž v nové verzi odstraněn. Následně jsem otestoval nové verze zásuvných modulů do KNOT WIKI. Zde jsem přišel na pár úprav, které musejí být po aktualizaci provedeny pro správnou funkčnost. Tyto změny a další užitečné poznámky k mému programu jsem umístil do souboru `poznamky.md`, který je umístěn ve složce s dokumentací programu. V tomto souboru se také nachází manuál pro aktualizaci WIKI na novou verzi a aktualizaci hlaviček stránek na moji novou verzi.

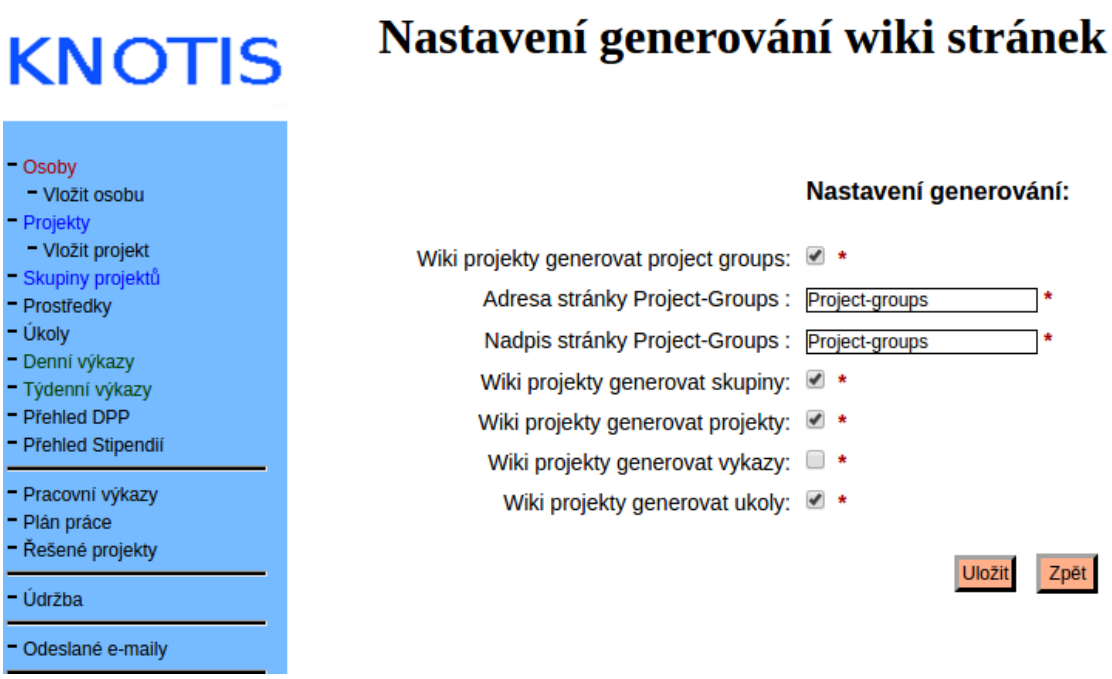
Po přípravě prostředí jsem začal s implementací a nejprve jsem upravil stránku pro nastavení generování WIKI, která se nachází v informačním systému KNOTIS, aby odpovídala novému návrhu. Tato upravená stránka je vidět na obrázku 6.1.

Dále jsem přidal kolonku pro externí URL do formuláře pro projekty a skupiny projektů a přidal zaškrtávací políčka pro vypnutí generování úkolů a výkazu do formuláře pro řešitele.

Po těchto nezbytných úpravách jsem přikročil k implementaci programu, který je spouštěn periodicky jednou denně a vytváří speciální hlavičky na soukromé WIKI výzkumné skupiny KNOT, jak bylo zmíněno v kapitole 2.2 a 5.2. Ukázka takové hlavičky je vidět na obrázku 6.7

6.1 Struktura programu

Program byl implementován pomocí objektového návrhu. Jeho struktura je vidět na obrázku 5.3. Tento návrh i samotná implementace odpovídá diagramům aktivit, popsaným v minulé kapitole. Každý modul je reprezentován vlastní třídou a submoduly jsou reprezentovány metodami tříd, přičemž existuje jedna abstraktní třída, která obsahuje metody, které jsou využity ve více modulech. Každá instance třídy, která reprezentuje



Obrázek 6.1: Ukázka nových možností pro nastavení automatického generování dokumentace v informačním systému KNOTIS.

modul, má k dispozici pole instancí ostatních tříd. Toto pole se předává objektu v jeho konstruktoru. Díky tomuto může objekt využívat veřejných metod ostatních objektů. Jednotlivé třídy a důležité soubory budou popsány dále v textu. Nastavení programu je umístěno v souboru `nastaveni.php`. Tam jsou umístěny texty poznámek, nastavení parametrů kontroly a tagy WIKI MarkUp. Dále metody v programu používají deklarace typů předávaných parametrů a návratových typů funkcí.

Struktura zpracování chyb

PHP je dynamicky typovaný jazyk, což zjednodušeně znamená, že programátor nemusí udávat typ proměnné a tento typ je odvozen automaticky z hodnoty dané proměnné. Tato vlastnost může být nebezpečná, pokud nějaká funkce očekává na vstupu určitý datový typ a dostane chybou programátora nebo špatnými daty jiný typ. Toto vede k pádu programu nebo hůře k jeho nepředvídatelnému chování. Abych se tomuto vyhnul a zpřehlednil program, tak jsem využil nové funkce jazyka PHP a to deklaraci návratových typů, deklaraci typů parametrů funkce a zachytávání těchto chyb pomocí bloků `catch(TypeError){}`. Toto samotné ovšem nestačí, proto jsem v programu použil kaskádové zpracování chyb pomocí bloků `try{}` a `catch(){}` pro zpracování výjimek. To znamená, že když nastane chyba, tak se program neukončí, což by bylo velice nepraktické, ale vyvolá se výjimka, která chybu zpracuje a vrátí řízení programu společně s informací o chybě zpět metodě, která tuto metodu volala. A ta se podle toho zachová.

page | discussion | edit | history | delete | move | protect | unwatch

Automatická podpora tvorby dokumentace projektů (wiki_z_knotis) [edit]

Vedoucí: Ing. Jaroslav Dytrych

Řešitel: Daniel Vosahlo (řeší od: 09.08.2016)

Zahájení projektu: 09.08.2016

Ukončení projektu: -

Typ projektu: Obecný projekt a BP

Skupina: misc

Redmine: http://knot.fit.vutbr.cz/redmine/projects/knotis

Prostředky:

název	cesta	oprávnění	patří k projektu	server
knotis	/mnt/minerva1/nlp/repositories/knotis/	zápis		minerva1
wiki_z_knotis	/mnt/minerva1/nlp/projects/wiki_z_knotis	zápis	Automatická podpora tvorby dokumentace projektů	minerva1
wiki_z_nlpis	/mnt/minerva1/nlp/projects/wiki_z_nlpis	čtení	Automatické generování wiki stránek z e-mailů a IS	minerva1

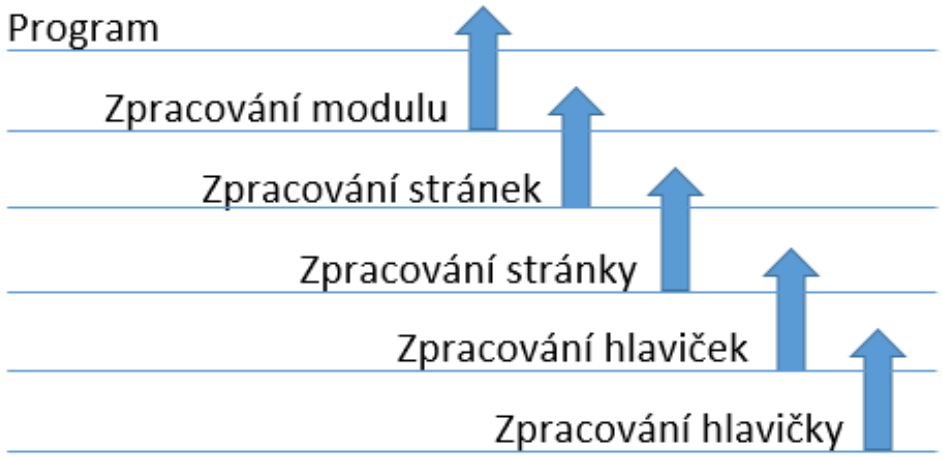
*Poznámka k oprávnění s *:* dříve bylo oprávnění **zápis**

Úkoly: Adresa úkolů projektu

Výkazy: Adresa výkazů projektu

Obrázek 6.2: Ukázka speciální hlavičky, která funguje jako záhlaví Wiki stránky. Tato hlavička patří jednomu z projektů řešených výzkumnou skupinou KNOT a zobrazuje základní informace o tomto projektu. Hlavička byla vytvořena programem, který je výsledkem této práce.

Tento princip je vidět na obrázku 6.3. Kromě chyby typu předávaného argumentu můžou v programu vzniknout i jiné chyby, tyto budou popsány v kapitole 6.9.



Obrázek 6.3: Ukázka kaskádového zpracování chyb ve třídách tohoto programu kontrolujících WIKI stránky.

6.2 Spuštění programu

Spouštěcím souborem celého programu je soubor `wiki_z_knotis.php`. Tento soubor načítá všechny ostatní potřebné soubory. Není potřeba přidávat načítání každé třídy ručně, protože je zde použita funkce `automatickeNacitaniTrid($trida)`, která načítá každou použitou třídu v programu automaticky. Stačí, když se třída bude jmenovat stejně jako soubor, ve kterém je umístěna. V tomto souboru se také nastavuje jazykové prostředí programu. Toto je důležité pro funkce, které řadí data, protože jinak tyto funkce neumí použít českou diakritiku a řadí data špatně. Poté je vytvořena instance třídy `KontrolaWiki`, což je hlavní třída celého programu a v konstruktoru jsou jí předány údaje pro přihlášení do databáze, WIKI a adresa KNOT WIKI, ze které program odstranil `HTTP/s`.

6.3 Třída KontrolaWiki

Instance této třídy funguje jako továrna (factory) a dědí abstraktní třídu *Kontrola*. Ve svém konstruktoru vytvoří instance třídy pro práci s databází, instanci třídy pro práci s WIKI API, instanci třídy `chyby` a instanci třídy `statistika` pro celkovou statistiku programu.

Hlavní metodou této třídy je metoda `kontrola()`. Ta nejprve spustí celkovou statistiku programu a pak provede přihlášení do databáze a WIKI. Poté načte WIKI výjimky a WIKI náhrady. WIKI náhrady jsou regulární výrazy, které nahrazují v dokumentaci některá slova. Metoda zjistí, jestli je nastavený příznak přechodu na novou verzi generování WIKI a pokud je, spustí metodu `_upgradujDb()` pro upgrade databáze.

Následuje spuštění všech modulů, které jsou zapnuté v informačním systému KNOTIS, jak bylo vidět na obrázku 6.1. K zjištění, jestli je modul zapnutý slouží metoda `_mameKontrolovat()`. Pokud je zapnutý, vytvoří se instance jeho třídy a v konstruktoru se jí předají data jako například WIKI výjimky, WIKI náhrady, instance ostatních tříd a další potřebné. Přidání nového modulu je jednoduché, stačí jej přidat na jakékoli místo do metody *kontrola* a přidat možnost do metody *mameKontrolovat*. Pokud uživatel bude chtít zobrazit statistiku svého modulu v závěrečné zprávě o kontrole, tak jeho modul musí vracet pole statistik vytvořené instancí třídy `Statistiky()`.

6.4 Třída pro práci s databází

Soubor `Db.php` obsahuje stejnojmennou třídu, která obstarává veškerou komunikaci s databází. Tato třída nedědí z žádné jiné třídy. Obsahuje instanční proměnné pro připojení k databázi a nastavení tohoto připojení. Pro komunikaci s databází se používá ovladač `PDO` s využitím předpřipravených dotazů, což znamená, že místo hodnot se do dotazu doplní „?“ a tyto hodnoty se pak předají v samostatném poli.

Pro pohodlnější práci s dotazy jsem implementoval metody, které jsou schopny databázový dotaz sestavit za uživatele. Implementace těchto metod je inspirována tutoriálem pana Davida Čapka [15]. Stačí si vybrat metodu pro vložení, editaci, výběr, atd. a předat jí asociativní pole s daty a parametry dotazu. Pokud chce uživatel sestavit dotaz sám, má k dispozici funkci `dotaz()`, která jako parametry bere sestavený dotaz a pole parametrů dotazu.

6.5 Třída Kontrola

Abstraktní třída *Kontrola* je obsažena v souboru `Kontrola.php` a jsou v ní definovány proměnné, které využívají všechny třídy, které z této třídy dědí. Díky tomuto máme zaručeno konstantní pojmenování instančních proměnných ve všech třídách. Dále třída obsahuje metody pro práci s textem, výpočet hashe, metody reprezentující submoduly, které jsou využity ve více modulech, a další metody využívané více třídami. Pokud by uživatel chtěl přidat nějaký další modul, který generuje nebo upravuje WIKI dokumentaci, tak by měl využít této třídy a jejích metod pro udržení přehlednosti a konzistence programu.

6.6 Třída Vyjimka

Třída `Vyjimka` se nachází v souboru `Vyjimka.php` a vytváří speciální výjimku. Tato výjimka se používá ve všech třídách programu pro odlišení důvodů vzniků výjimek. Některé metody mohou vyvolat 2 druhy výjimek. Klasickou výjimku `Exception`, nebo naší třídu `Vyjimka`. Díky tomuto může program rozlišit, jestli se například jedná o chybu kontroly, nebo jenom o přítomnost tagu negenerovat, který nám říká, že se stránka nemá kontrolovat a místo toho se má přeskočit. Funkce každé výjimky je popsána v dané metodě.

6.7 Třída Statistiky

Soubor `Statistiky.php` obsahuje stejnojmennou třídu pro vedení statistik modulu i programu. Statistika se zahájí voláním metody `start()`, která uloží čas do instanční proměnné objektu. Dále obsahuje metody pro inkrementaci počtu hlaviček a zaznamenání doby jejich zpracování. Pro ukončení statistiky stačí zavolat metodu `stop()`. Ta vypočte čas, jak dlouho byla statistika aktivní, zjistí největší využití paměti po dobu měření této statistiky a tyto údaje vrátí jako pole.

Pokud chce uživatel přidat další měřenou statistiku, přidá metody pro její zpracování a přidá index do pole pro vrácení. Pokud chce svoje nové statistiky pro různé moduly sčítat a zobrazit součet v celkové statistice, musí na konci metody `kontrola()` v souboru `KontrolaWiki.php` přidat podmínku, aby cyklus sčítal i jeho novou statistiku. Nakonec musí přidat její vypsání a to bude popsáno v kapitole 6.9.

6.8 Třída WikiApi

Tato třída se stará o komunikaci mezi programem a Media Wiki API. Od nových verzí Media Wiki se silně doporučuje, aby se při komunikaci s API posílaly speciální hlavičky HTTP User Agent. Tato hlavička je vytvořena pomocí metody `_userAgent()`. Metoda používá pro posílání a zpracování požadavků knihovnu *Libcurl*.

Nejprve je nutné se přihlásit do WIKI, což se provádí metodou `prihlas()`, která nejprve zkontroluje, jestli je přítomný soubor pro uložení dočasných dat sezení a jestli má program právo do tohoto souboru zapisovat. Poté vytvoří hlavičku User Agent a uloží ji do instanční proměnné, aby byla přístupná ostatním metodám a ty ji nemusely vytvářet znovu. Dále metoda obsahuje metody pro zjištění existence stránky, smazání stránky, vytvoření stránky, editaci obsahu stránky, vrácení obsahu stránky a odhlášení z WIKI API.

Pokud metoda chce upravovat stránku, musí využít speciálního postupu. Nejprve pošle požadavek na API, aby jí vytvořilo takzvaný *editační token*. Pokud jsme API poslali správná data, tak nám tento token vytvoří. Poté stačí odeslat další požadavek, do kterého umístíme tento token, a API na základě tohoto požadavku upraví požadovanou stránku a vrátí odpověď, jestli se operace podařila. Tento postup se opakuje při každé změně stránky.

6.9 Třída Chyby

V kapitole 6.1 byl popsán způsob zpracování chyb. Kromě chyb typů argumentů jsem v programu implementovat i vlastní chyby. Ty můžou být několika druhů:

- *Varování* je něco jako upozornění, které se zaznamená, ale běh programu nijak neovlivní.
- *Chyba skriptu* znamená, že se někde v metodě stala chyba, metoda byla ukončena a řízení bylo předáno jiné metodě.

- *Chyba při kontrole obsahu* – tato chyba uživateli říká, že nastala chyba při kontrolování obsahu stránky.
- *Nelze vytvořit stránku* znamená, že program nemohl vytvořit nebo editovat stránku na WIKI.

Po těchto chybách se opět ukončuje metoda a předává řízení. Posledním druhem chyby je:

- *Kritická chyba* – pokud nastane tato chyba, program končí. Toto se může stát například při nedostupnosti WIKI API, nebo databáze.

Situace, kdy nastanou tyto chyby, je možné vidět v diagramech aktivit v příloze A.

Zpracování chyb má na starosti třída `Chyby` v souboru `Chyby.php`. Pokud nastane nějaká chyba kromě kritické, program zavolá příslušnou metodu, která chybu zpracuje, dá do správného formátu a uloží do pole chyb. Pro formátování chyb se používá metoda `_formatujChybu()`. Ta z objektu výjimky získá důležitá data, naformátuje je a přidá text chyby, který se předává společně s objektem výjimky v argumentech metody.

Před ukončení programu se zavolá funkce `zpravaOKontrole()`, která vytvoří zprávu o průběhu programu, a pokud jsou přítomny chyby, tak je vloží do informačního systému KNOTIS. Toto je vidět ve spodní části diagramu průběhu programu A.1. Zpráva o průběhu programu je posílána jako e-mail správci programu, jehož adresa je umístěna v nastavení programu v souboru `Nastaveni.php`. Tento e-mail je uložen do databáze, takže pokud odeslání e-mailu selže, data se neztratí.

Zpráva o průběhu programu obsahuje statistiku jednotlivých modulů a také celkovou statistiku programu. V dolní části jsou přítomny sekce s chybami. Každý typ chyby má svou sekci. O formátování zprávy se stará metoda `_sestaveniEmailu()`. Pokud uživatel chce přidat nějakou svoji statistiku a chce, aby se vyskytovala ve zprávě o kontrole, tak musí přidat řádek který obsahuje název indexu pole a jeho popis. V ukázce 6.1 je vidět část e-mailu obsahující zprávu o kontrole a sekce s chybami.

Výpis 6.1: Ukázka podoby výpisu chyb v e-mailu se zprávou o průběhu programu.

```
celkovaStatistikaSkriptu :
Skript začal 14:34:45 - 14.05.2017, skončil 14:34:45 - 14.05.2017
a trval 1,3138678471247 minut.
Největší memory usage: 6,38 MB
Počet zpracovaných stránek: 962;
Počet stránek přeskočených, kvůli vyjímkám: 6;
Počet stránek přeskočených, kvůli tagu negenerovat: 0;
Počet přeskočených hlaviček (heshe se ==): 2353;
Vytvořeno nových stránek: 0;
Počet editovaných stránek: 962

Varování skriptu:
Varování při běhu programu a seskupování URL
na stránce : Https://knot.fit.vutbr.cz/
```

```
Informace o chybě: Máme odkaz, který leží mimo WIKI
V souboru:
[/var/www/html/wiki_z_knotis/KNOTIS-WIKI/Kontrola.php Line:1532]
Datum chyby: 14:34:46 - 14.05.2017

Varování při běhu programu a seskupování URL
na stránce : Https://knot.fit.vutbr.cz/
Informace o chybě: Máme odkaz, který leží mimo WIKI
V souboru:
[/var/www/html/wiki_z_knotis/KNOTIS-WIKI/Kontrola.php Line:1532]
Datum chyby: 14:34:46 - 14.05.2017
```

6.10 Třída ProjectGroups

Tato třída je obsažena v souboru `ProjectGroups.php`. V konstruktoru této třídy se předají a uloží informace důležité pro kontrolu jako WIKI výjimky, náhrady, adresa stránky Project Groups a její nadpis. Pokud je v KNOTIS tento modul zapnutý, ale není nastaven nadpis nebo adresa, tak se tyto informace vezmou z předdefinovaných hodnot z nastavení programu. Adresa stránky se v konstruktoru sestaví dohromady. Vezme se adresa WIKI, přidá se adresa API a na závěr adresa stránky. Toto se děje proto, že kdyby administrátor WIKI chtěl změnit adresu API, stačí mu změnit koncovku v nastavení. Tento postup je využit ve všech třídách, které pracují s WIKI.

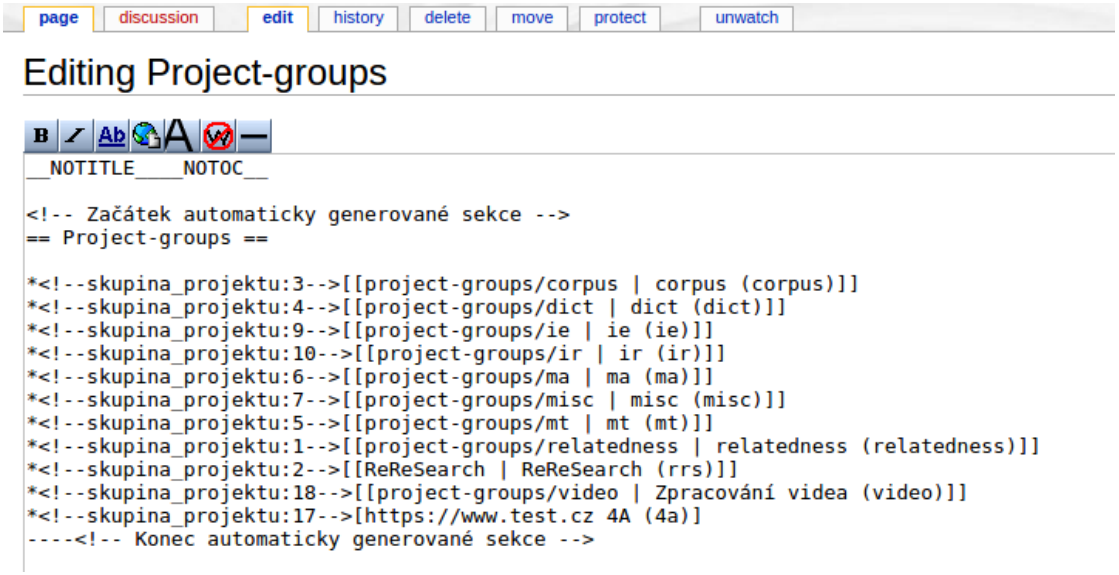
Hlavní metodou třídy je metoda `kontrola()`. Ta nejprve zkontroluje, jestli nastavená adresa není ve výjimce a pokud není, zkontroluje existenci této stránky. Třída po ukončení své kontroly vrací pole statistik.

Pokud stránka neexistuje, tak se volá metoda `_vytvorStranku()`, která vytvoří stránku, vypočítá hashe a následně je uloží do databáze. Pokud stránka existuje, zavolá se metoda `_zkontrolujObsahStranky()`. Tato metoda pak provede kontrolu stránky, výpočet a uložení hashů, jak je znázorněno v diagramu Project Groups A.3. Obě metody před uložením hashů nejprve zkontrolují, jestli vypočtený hash nemá stejnou hodnotu jako starý. Pokud je hash stejný jako minulý, tak je z pole hashů k uložení do databáze odstraněn.

Pro ukládání hashů jsem implementoval postup, který nejprve načte identifikátory hashů, které jsou přítomny v databázi, a porovná je s identifikátory hashů, které se mají vložit do databáze. Díky tomuto postupu program zjistí, jestli má do databáze tento hash vložit, nebo jenom změnit jeho hodnotu. Tento postup je implementován ve všech třídách, které ukládají hashe do databáze.

Ukázku struktury hlavičky Project Groups můžeme vidět na obrázku 6.4. Jako první se na řádku nachází tag s informací, že se jedná o skupinu projektů a také je v něm obsažen identifikátor skupiny. Poté je umístěn element WIKI MarkUp pro vnitřní nebo vnější odkaz. Vnitřní odkaz se skládá z odkazu, který může obsahovat pouze cestu ke stránce bez adresy WIKI, a aliasu, který bude na WIKI zobrazen místo odkazu. Alias se v tomto případě skládá ze jména skupiny a zkratky skupiny. Program počítá i s variantou, že od-

kaz na skupinu bude vést mimo WIKI. V tomto případě se použije Wiki MarkUp element pro vnější odkaz, který obsahuje celou adresu a alias. Toto můžeme vidět na posledním řádku v odkazu na skupinu *4A*.



Obrázek 6.4: Ukázka struktury hlavičky Project Groups.

6.11 Třída SkupinyProjektu

Třída je obsažena v souboru `SkupinyProjektu.php`. V konstruktoru jsou třídě předány potřebné informace. Tentokrát se ale vytvoří adresa pouze z adresy WIKI a API. Adresy stránek se budou v průběhu kontroly měnit. Hlavní metodou této třídy je metoda `kontrola()`.

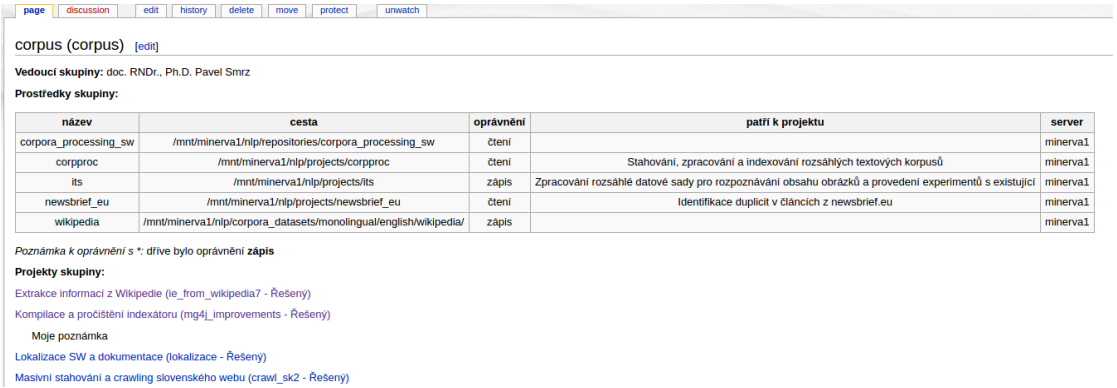
Nejprve se načtou základní data a poté se seskupí adresy podle URL, jak bylo popsáno dříve v návrhu 5.2. Seskupení je realizováno pomocí metody `seskupPodleUrl()` v souboru `Kontrola.php`. Ta nejprve dekóduje odkazy URL, pro udržení konzistence. Poté převede počáteční písmeno každé WIKI adresy na velké. Systém Media Wiki s adresou dělá přesně to stejné, takže pro něj adresa *Adresa* bude stejná, jako *adresa*, ale *AdreSa* už se liší. Toto by ale způsobovalo chyby v seskupení, proto se provádí převedení prvního písmena na velké. Po seskupení vznikne dvoudimenzionální pole, kde indexem je adresa URL a hodnotou je pole identifikátorů skupin. Tato metoda také vyseparuje odkazy URL, které vedou mimo WIKI a vytvoří varování.

Po seskupení metoda pomocí cyklu zpracovává jednotlivé URL. Poté následuje kontrola na přítomnost ve WIKI výjimce, načtení dat pro skupiny na dané URL a zpracování stránky s danou URL.

Nově navržená struktura hlaviček se dimenzionálně liší od té staré a je potřeba převést starou strukturu hlaviček na novou. Proto je dalším krokem programu zjiš-

tění, jestli není nastaven mód pro aktualizaci WIKI. Pokud ano, zavolá se metoda `_prechodZeStareVerze()`. Problémem bylo, že staré hlavičky byly často velmi nekonzistentní. Metoda také ví, že se na stránce skupiny mohla vyskytovat pouze skupina. Proto metoda nejprve uloží text pod automaticky generovanou sekcí a poté odstraní hlavičku a ze staré hlavičky si vezme pouze obsah tabulky prostředků, která obsahuje i prostředky, které již neexistují. Tyto prostředky jsou uloženy do pole a toto je pak předáno metodě pro vytvoření stránky místo dat prostředků z databáze. Ta na stránce vytvoří novou hlavičku a pod ni přidá zbytek stránky. Tyto nové hlavičky ale nemusejí být aktuální. Proto se musí program spustit dvakrát. Jednou s modem aktualizace a podruhé normálně. Tím je zajištěna aktuálnost a bezproblémový přechod na novou verzi generování.

Když není mód přechodu na novou verzi zapnut, provede se kontrola stránky pomocí metody `_zkontrolujObsahStranky()`. Jak bylo popsáno v návrhu, tak nově může být na jedné stránce více hlaviček skupin projektů a projektů zároveň. Toto jsem implementoval pomocí postupu, který postupně načítá jednotlivé hlavičky zpracuje je, a poté je ukládá do separátních polí a indexuje podle identifikátoru, který se vytvoří z jejich názvu a zkratky, takže je potom možné hlavičky seřadit podle abecedy. Po dokončení kontroly se seřadí hlavičky skupin podle abecedy, dále se hlavičky seřadí v pořadí skupiny, projekty, přidá se zbytek stránky a stránka se uloží.


page discussion edit history delete move protect unwatch

corpus (corpus) [edit]

Vedoucí skupiny: doc. RNDr., Ph.D. Pavel Smrz

Prostředky skupiny:

název	cesta	oprávnění	patří k projektu	server
corpora_processing_sw	/mnt/minerva1/nlp/repositories/corpora_processing_sw	čtení		minerva1
corpproc	/mnt/minerva1/nlp/projects/corpproc	čtení	Stahování, zpracování a indexování rozsáhlých textových korpusů	minerva1
its	/mnt/minerva1/nlp/projects/its	zápis	Zpracování rozsáhlé datové sady pro rozpoznávání obsahu obrázků a provedení experimentů s existující	minerva1
newsbrief_eu	/mnt/minerva1/nlp/projects/newsbrief_eu	čtení	Identifikace duplicit v článcích z newsbrief.eu	minerva1
wikipedia	/mnt/minerva1/nlp/corpora_datasets/monolingual/english/wikipedia/	zápis		minerva1

*Poznámka k oprávnění s *:* dříve bylo oprávnění **zápis**

Projekty skupiny:

Extrakce informací z Wikipedie (ie_from_wikipedia7 - Řešený)

Kompilace a pročištění indexátoru (mg4j_improvements - Řešený)

Moje poznámka

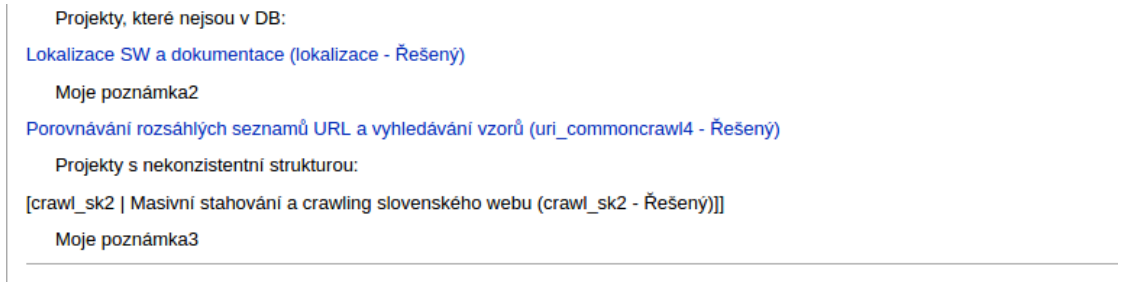
Lokalizace SW a dokumentace (lokalizace - Řešený)

Masivní stahování a crawling slovenského webu (crawl_sk2 - Řešený)

Obrázek 6.5: Ukázka části hlavičky skupiny projektů.

Ukázka části hlavičky skupiny projektů je vidět na obrázku 6.5. Jak bylo řečeno v návrhu, uživatel může přidat svoji poznámku pod odkaz na projekt, jak je také možno vidět na obrázku 6.5. Zde byla otázka, jak zachovat poznámku, pokud projekt změní stav nebo bude odstraněn z databáze, nebo pokud uživatel poničí jeho strukturu. Pokud byla struktura projektu poničena, tak by poznámka měla u tohoto projektu zůstat. Uživatel ji kdykoli může přesunout k opravenému projektu nebo odstranit. Toto jsem vyřešil tím, že se projekty ukládají do multidimenzionálních polí podle jejich typu. Jako je pole řešených projektů, pole projektů navíc, projektů s poničenou strukturou, pole nezadaných projektů, atd. Po uložení projektu do daného pole se zachová v proměnné jeho index a nastaví se speciální identifikátor vložení, který identifikuje pole. Pokud program narazí na poznámku, tak podle identifikátoru nejprve vybere pole a poté podle

indexu posledního projektu vybere index, který obsahuje pole s projektem, kterému do položky poznámky, která obsahuje pole poznámek přidá načtenou poznámku. Díky tomuto způsobu je zaručeno, že se načtou všechny poznámky projektu a uživatel jich může přidat více. Toto platí i když je projekt poničený, nebo odstraněn z databáze jak je vidět na obrázku 6.6.



Obrázek 6.6: Ukázka zachování poznámek u projektu, ke kterému patří v hlavičce skupiny projektů.

6.12 Třída Projekty

Třída pro kontrolu projektů je umístěna v souboru `Projekty.php`. Platí u ní stejný postup kontroly jako u kontroly skupin projektů. Nejprve se načtou URL a identifikátory všech projektů, které mají zapnuté generování. Tyto projekty se seskupí podle URL do pole. Z pole se odstraní URL, které jsou ve WIKI výjimkách. Poté se pole projektů rozdělí pomocí cyklu a funkce `array_slice()` na více částí. Tyto části se uloží do pole projektů ke kontrole. Velikost částí se nastavuje v nastavení skriptu. Po tomto kroku se načtou podrobná data hlaviček pro adresy v bloku a začne kontrola adres obsažených v každé části pole projektů ke kontrole. Tato implementace zajišťuje, že modul přistoupí ke každé stránce jen jednou a nepracuje s velkým množstvím dat. Po tomto následuje kontrola existence stránky. Pokud stránka existuje, následuje kontrola jestli není zapnutý mód aktualizaci, a pokud není, následuje kontrola stránky. Případně pokud neexistuje, tak její vytvoření.

Implementace aktualizace hlaviček projektů je složitější než u skupin projektů. Toto je způsobeno faktem, že stará verze dovolovala umístění více hlaviček projektů na jedné adrese. Dalším faktem byl někdy až katastrofální stav hlaviček, kdy chyběly identifikátory, nebo byly přesunuty až na konci hlavičky a byla poničena struktura hlavičky. Toto jsem vyřešil tak, že metoda nejprve prochází hlavičky a kontroluje, jestli se v nich nachází identifikátor. Pokud nenachází, zaznamená chybu, a pokud se v ní nachází na špatném místě, tak ho přesune na začátek hlavičky projektu. Díky tomuto jsem získal seznam 3 adres na kterých se nacházejí hlavičky, které jsou tak poničené, že se musí před aktualizací ručně opravit. Pokud je po tomto kroku hlavička v dobrém formátu, tak metoda vyjme data tabulky prostředků a uloží do pole prostředků. Až skončí kontrola hlaviček, tak je automaticky generovaná sekce vymazána a stejně jako v případě skupin se zavolá

funkce pro vytvoření stránky s načtenými daty, která jí vytvoří znovu. A na závěr se pod ni přidá zbytek stránky, který jsme si před upgradem uložili do proměnné.

Pro vytvoření stránky se používá metoda `_vytvorStranku()` a pro editaci se používá metoda `zkontrolujObsahStranky()`. Výsledek těchto metod můžeme vidět na obrázku 6.7, který ukazuje hlavičku projektu při zapnutí generování a přítomnosti všech informací.


page discussion edit history delete move protect unwatch

Automatická podpora tvorby dokumentace projektů (wiki_z_knotis) [edit]

Vedoucí: Ing. Jaroslav Dytrych

Řešitel: Daniel Vosahlo (řeší od: 09.08.2016)

Zahájení projektu: 09.08.2016

Ukončení projektu: -

Typ projektu: Obecný projekt a BP

Skupina: misc

Redmine: http://knot.fit.vutbr.cz/redmine/projects/knotis

Externí URL: http://www.fit.vutbr.cz/

Prostředky:

název	cesta	oprávnění	patří k projektu	server
knotis	/mnt/minerva1/nlp/repositories/knotis/	zápis		minerva1
wiki_z_knotis	/mnt/minerva1/nlp/projects/wiki_z_knotis	zápis	Automatická podpora tvorby dokumentace projektů	minerva1
wiki_z_nlpis	/mnt/minerva1/nlp/projects/wiki_z_nlpis	čtení	Automatické generování wiki stránek z e-mailů a IS	minerva1

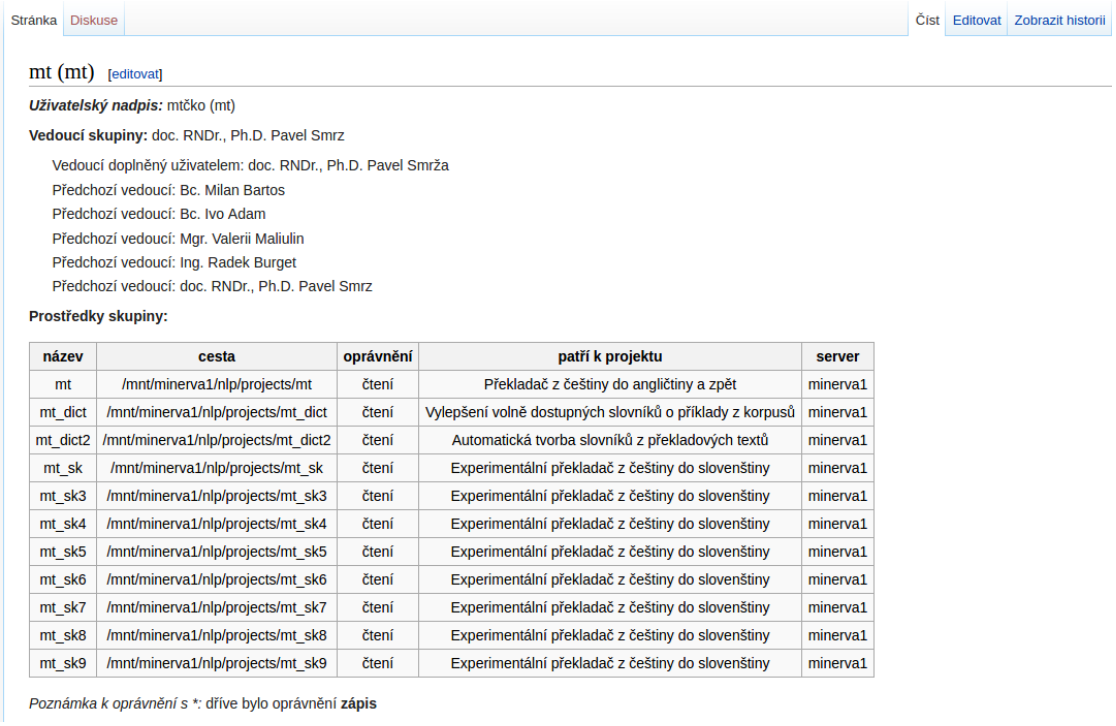
*Poznámka k oprávnění s *:* dříve bylo oprávnění **zápis**

Úkoly: Adresa úkolů projektu

Výkazy: Adresa výkazů projektu

Obrázek 6.7: Ukázka hlavičky projektu.

Kromě tabulky prostředků se ke každému řádku může vytvořit poznámka o uživatelské změně a uživatel také může ke každému řádku přidat svoji poznámku, jak bylo popsáno v návrhu a je vidět v diagramech. Na obrázku 6.8 je vidět hlavička skupiny projektů s historií vedoucích. Jak je zřejmé, program musí hlídat, aby se hlavička zbytečně nezahlcovala, například stejnými poznámkami. Pro řešení tohoto problému jsem implementovat metodu `mameStejnouPoznamku()`, která jako parametry přijímá pole poznámek a novou poznámku. Metoda nejprve zajistí, aby se v textu poznámky mezi slovy vyskytovala maximálně 1 mezera, a dále ořízne mezery ze začátku a konce textu. Po těchto úpravách zkontroluje, jestli se tato poznámka v předaném poli již nenachází a pokud ano, tak ji zahodí. Metoda vrací pole poznámek a je použita ve všech třídách kontrolující hlavičky WIKI.



Stránka Diskuse Číst Editovat Zobrazit historii

mt (mt) [editovat]

Uživatelský nadpis: mtčko (mt)

Vedoucí skupiny: doc. RNDr., Ph.D. Pavel Smrz

Vedoucí doplněný uživatelem: doc. RNDr., Ph.D. Pavel Smrža
Předchozí vedoucí: Bc. Milan Bartos
Předchozí vedoucí: Bc. Ivo Adam
Předchozí vedoucí: Mgr. Valerii Maliulin
Předchozí vedoucí: Ing. Radek Burget
Předchozí vedoucí: doc. RNDr., Ph.D. Pavel Smrz

Prostředky skupiny:

název	cesta	oprávnění	patří k projektu	server
mt	/mnt/minerva1/nlp/projects/mt	čtení	Překladač z češtiny do angličtiny a zpět	minerva1
mt_dict	/mnt/minerva1/nlp/projects/mt_dict	čtení	Vylepšení volně dostupných slovníků o příklady z korpusů	minerva1
mt_dict2	/mnt/minerva1/nlp/projects/mt_dict2	čtení	Automatická tvorba slovníků z překladových textů	minerva1
mt_sk	/mnt/minerva1/nlp/projects/mt_sk	čtení	Experimentální překladač z češtiny do slovenštiny	minerva1
mt_sk3	/mnt/minerva1/nlp/projects/mt_sk3	čtení	Experimentální překladač z češtiny do slovenštiny	minerva1
mt_sk4	/mnt/minerva1/nlp/projects/mt_sk4	čtení	Experimentální překladač z češtiny do slovenštiny	minerva1
mt_sk5	/mnt/minerva1/nlp/projects/mt_sk5	čtení	Experimentální překladač z češtiny do slovenštiny	minerva1
mt_sk6	/mnt/minerva1/nlp/projects/mt_sk6	čtení	Experimentální překladač z češtiny do slovenštiny	minerva1
mt_sk7	/mnt/minerva1/nlp/projects/mt_sk7	čtení	Experimentální překladač z češtiny do slovenštiny	minerva1
mt_sk8	/mnt/minerva1/nlp/projects/mt_sk8	čtení	Experimentální překladač z češtiny do slovenštiny	minerva1
mt_sk9	/mnt/minerva1/nlp/projects/mt_sk9	čtení	Experimentální překladač z češtiny do slovenštiny	minerva1

*Poznámka k oprávnění s *:* dříve bylo oprávnění **zápis**

Obrázek 6.8: Tento obrázek ukazuje historii vedoucích a poznámku o ruční uživatelské změně vedoucího.

Projekty se od skupin projektů liší také v možnosti zvolit různý stupeň generování o informací projektu. Drobnou komplikací byl způsob, jak zvládnout situaci, kdy se projektu nastaví generování celé hlavičky, potom se nastavení změní na generovat pouze nadpis, a pak se třeba po nějaké době zase změní zpět na generovat hlavičku. V tomto případě metoda kontroly zjistí, jestli se v hlavičce projektu vyskytuje nějaká poznámka. Pokud se vyskytuje, tak ví, že hlavička obsahuje uživatelský obsah a zkontroluje pouze nadpis a zbytek hlavičky ignoruje. Jediná poznámka, která se nebere jako uživatelský obsah, je poznámka s předchozím vedoucím.

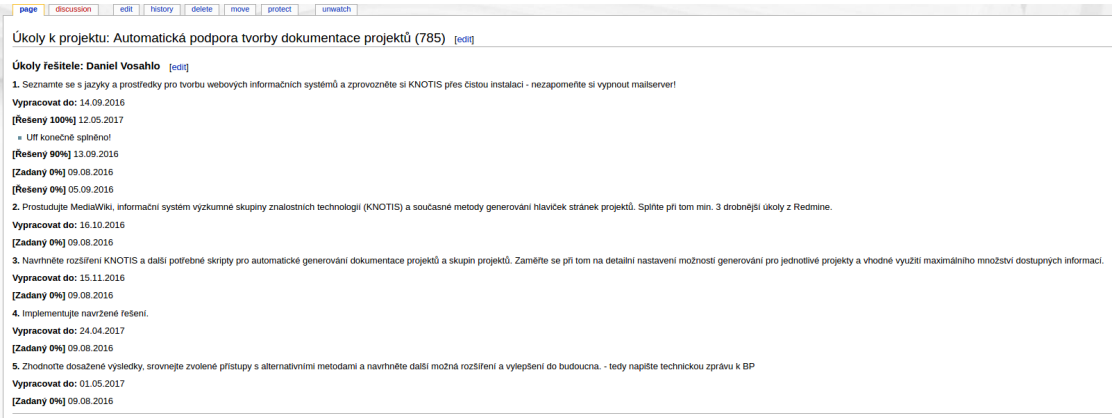
6.13 Třída Ukoly

Třída se nachází v souboru `Ukoly.php` a má stejnou strukturu jako třída pro kontrolu projektů a skupin projektů. Kromě metody pro aktualizaci, která zde není potřeba, protože starý program nikdy úkoly k projektům negeneroval. Ukázku stránky s úkoly k projektu můžeme vidět na obrázku 6.9.

Stránka obsahuje automaticky generovanou sekci, která na svém začátku obsahuje nadpis s názvem a identifikátorem projektu. Dále následují hlavičky řešitelů projektu, kteří mají zadaný nějaký úkol. Hlavička řešitele se skládá z nadpisu se jménem řešitele a hlaviček jednotlivých úkolů. Každá hlavička úkolu obsahuje pořadové číslo úkolu, text

úkolu, datum, do kdy je nutné úkol vypracovat, a tag stavu úkolu složený ze jména stavu a procentuálního splnění stavu. Stav také může obsahovat komentář řešitele z informačního systému KNOTIS, který je umístěn pod stavem.

Díky této struktuře jsem musel na rozdíl od ostatních tříd pro uložení hashů použít dvě tabulky. Jednu pro hashe nadpisů hlaviček řešitelů a druhou pro hashe hlaviček úkolů. Tuto implementaci jsmem zvolil kvůli faktu, že řešitel může mít 1..N úkolů a bylo by nešikovné opakovat v tabulce N krát data hashe nadpisu řešitele. Další věcí, která je zde implementována trochu jinak, je načítání poznámek. Uživatel totiž může v systému KNOTIS zadat poznámku ke stavu, která může obsahovat nové řádky. Poznámka je v databázi uložena jako jeden celek, ale na WIKI by ji program rozeznal jako poznámku a text navíc v hlavičce. Proto tato třída má upravený postup pro načítání poznámek, který když identifikuje poznámku, tak přeskakuje prázdné řádky a načítá text, dokud nenarazí na nějaké klíčové slovo hlavičky úkolu.


page discussion edit history delete move protect unwatch

Úkoly k projektu: Automatická podpora tvorby dokumentace projektů (785) [edit]

Úkoly řešitele: Daniel Vosahlo [edit]

1. Seznamte se s jazyky a prostředky pro tvorbu webových informačních systémů a zprovozněte si KNOTIS přes čistou instalaci - nezapomeňte si vypnout mailserver!
Vypracovat do: 14.09.2016
[Řešený 100%] 12.05.2017
- Uff konečně splněno!

[Řešený 90%] 13.09.2016
[Zadaný 0%] 09.08.2016
[Řešený 0%] 05.09.2016

2. Prostudujte MediaWiki, informační systém výzkumné skupiny znalostních technologií (KNOTIS) a současné metody generování hlaviček stránek projektů. Splňte při tom min. 3 drobnější úkoly z Redmine.
Vypracovat do: 16.10.2016
[Zadaný 0%] 09.08.2016

3. Navrhněte rozšíření KNOTIS a další potřebné skripty pro automatické generování dokumentace projektů a skupin projektů. Zaměřte se při tom na detailní nastavení možností generování pro jednotlivé projekty a vhodné využití maximálního množství dostupných informací.
Vypracovat do: 15.11.2016
[Zadaný 0%] 09.08.2016

4. Implementujte navržené řešení.
Vypracovat do: 24.04.2017
[Zadaný 0%] 09.08.2016

5. Zhodnoťte dosažené výsledky, srovnejte zvolené přístupy s alternativními metodami a navrhněte další možná rozšíření a vylepšení do budoucna. - tedy napište technickou zprávu k BP
Vypracovat do: 01.05.2017
[Zadaný 0%] 09.08.2016

Obrázek 6.9: Stránka úkolů k projektu s hlavičkou úkolů řešitele.

6.14 Třída MazaniProjektu

Tato třída slouží k mazání stránek projektů, které jsou mazány z informačního systému KNOTIS. Třída je umístěna v souboru `MazaniProjektu.php`. Tato třída není součástí běžné kontroly dokumentace a spouští se pouze při mazání projektu v KNOTIS. KNOTIS vytvoří instanci této třídy, která vytvoří instanci třídy pro práci s WIKI API, třídu pro vlastní výjimku a načte soubor s nastavením. Toto je také jediná třída, která je propojena s KNOTIS. Pro umožnění tohoto propojení jsem upravil soubor KNOTIS `projekty.php`. Musel jsem do souboru přidat cestu ke třídě, cestu k nastavení programu pro generování WIKI a logiku pro vytvoření třídy a zpracování jejího výstupu.

Po kontrole stránky projektu, které byla popsána v kapitole 5.2, vrátí třída KNOTIS pole s položkou status a položkou text chyby. Status může nabývat třech stavů:

- *0=>vše proběhlo v pořádku, stránka je smazána*
- *1=> Stránku nelze smazat, protože obsahuje uživatelský text*

- *3=>Nastala chyba při mazání stránky*

Text s chybou se posílá pouze pokud nastala chyba číslo 3. KNOTIS tyto informace zpracuje a informuje o nich uživatele.

6.15 Dokumentace programu

Dokumentace tohoto programu je generována programem PhpDoc a je umístěna ve složce `Dokumentace/Php_doc`. Každá metoda také obsahuje hlavičku se základními informacemi o dané metodě, jako je popis vstupů, výstupů a základní popis metody. Tyto informace jsou potřeba ke správnému vygenerování dokumentace. Ukázku takové dokumentace můžeme vidět v ukázce 6.2. Kód také obsahuje komentáře každé důležité konstrukce a pokud je to nutné, je v komentáři zobrazena struktura nějakého složitého pole.

Výpis 6.2: Ukázka dokumentace metody. Tento formát dokumentace je požadován programem PhpDoc pro automatické generování dokumentace.

```
/**
 * Metoda pro zjištění, jestli se jedná o vnitřní odkaz do wiki.
 * Odkaz porovnáme s odkazem na wiki, který jsme zbavili http/s
 * a www.
 *
 * @param string $odkaz : String s odkazem
 *
 * @global string $wikiProjektyRootOriznuto String s ořiznutou
 adresou wiki(bez http/s a www)
 *
 * @return bool Pokud odkaz obsahuje tento string, tak vrátíme
 true, jinak false
*/
    public function jednaSeOVnitrniOdkaz(string $odkaz):bool
    {
        global $wikiProjektyRootOriznuto;

        if (strpos($odkaz, $wikiProjektyRootOriznuto) === false) {
            return false;
        }
        return true;
    }
```

Program obsahuje 14 souborů a 12 tříd. Tyto soubory obsahují celkově 8046 řádků kódu a 4625 řádků s komentáři. Do Komentářů se počítají i řádky, na kterých se nachází kód s komentářem. Tyto čísla nezahrnují úpravy informačního systému KNOTIS a soubor s poznámkami k implementaci. K dispozici je i 26 diagramů aktivit, které znázorňují logiku programu a fungují jako dokumentace.

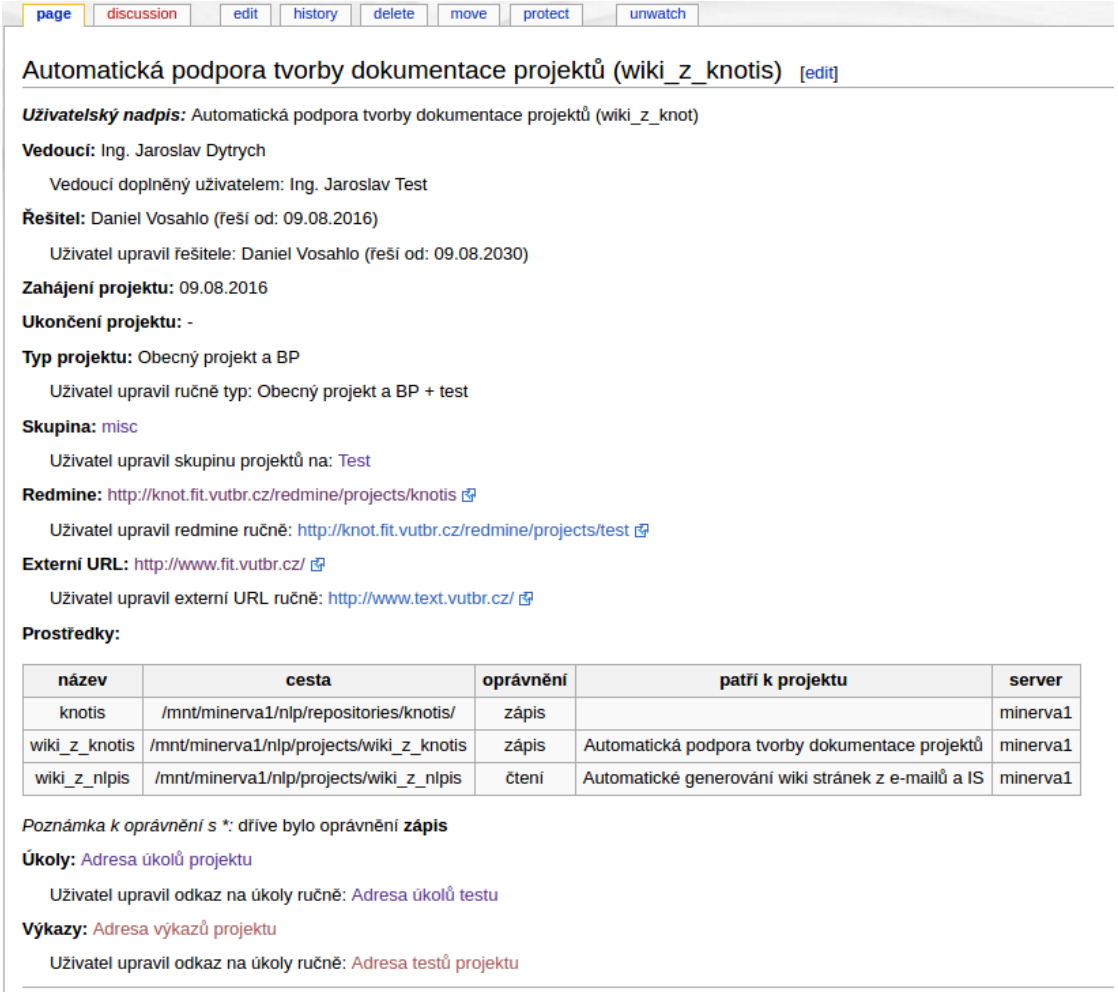
Kapitola 7

Testování

Jelikož tento program má nahradit starý program pro generování dokumentace a hlavním důvodem této náhrady je veliká chybovost starého programu, tak jsem prováděl extenzivní testování všech jeho funkcí. Testování probíhalo velmi pečlivě a průběžně po implementaci jednotlivých submodulů, modulů a na závěr se testoval běh celého programu a jeho výkonnost.

Otestoval jsem chování programu při chybných datech, nepřístupném WIKI API, databázi nebo chybě předávaného argumentu nějaké funkce. Největší pozornost jsem ale věnoval testování hlaviček a všech případů, co mohou nastat, jako je například uživatelská změna dat, vymazání některých řádků z hlavičky, záměrné poničení hlavičky, smazání hlaviček, více hlaviček skupin a projektů na jedné adrese, stránka s chybějící automaticky generovanou sekcí, přidání textu navíc do různých míst hlavičky, atd. Výsledky těchto a ostatních testů jsou zdokumentovány pomocí obrázků a uloženy ve složce `Dokumentace/Vysledky testu`.

Na obrázku 7.1 je vidět uživatelská změna dat, která byla provedena uživatelem v hlavičce projektu. Jediné řádky, kde se nedělají poznámky, jsou řádky se zahájením a ukončením řešení, protože se zde uživatelská změna nepřipouští.



Obrázek 7.1: Tento obrázek ukazuje uživatelskou změnu dat v hlavičce projektu, její rozpoznání a následné vytvoření poznámek.

7.1 Statistiky hotového programu

Tato kapitola bude obsahovat statistiku programu jako je počet zkontrolovaných projektů, doba trvání programu, největší využití paměti, ideální počet načtených dat, atd.

Nalezení správného počtu načtených projektů a úkolů.

Jak bylo zmíněno v kapitole o implementaci modulu Projekty a Úkoly, je nutné najít dobrý poměr načtených dat ku výkonu modulu. Pro tento účel jsem volil různá čísla N a sledoval statistiku využití paměti a doby běhu modulu. Do tabulky jsem pro přehlednost zvolil jen 5 nejlepších výsledků. Pro každou konfiguraci v tabulce bylo provedeno 5 pokusů a výsledné hodnoty jsou jejich váženým průmětem. Všechna testování výkonu

byla provedena na virtuální stroji běžícím na systému Linux Mint 18.1 s 4 GB přidělené paměti a procesorem Intel core I7 3770 3,4 GHz. Výsledky pro modul Projekty jsou vidět v tabulce 7.1 a vyšlo z nich, že ideálním počtem zpracovaných adres je číslo 50. Tabulka 7.2 ukazuje tuto statistiku pro modul Úkoly, kde ideální N vyšlo 200. Tyto hodnoty se mohou lišit systém od systému a jsou závislé na hardwarové výbavě testované sestavy.

Zvolené N	Doba trvání modulu (s)	Maximální využití paměti (MB)
200	29,10	3,51
100	29,65	3,26
50	28,10	3,11
25	28,80	3,05

Tabulka 7.1: Tabulka statistiky modulu pro kontrolu projektů, který kontroluje 502 hlaviček při různě zvolené velikosti bloku adres, které se mají kontrolovat.

Zvolené N	Doba trvání modulu (s)	Maximální využití paměti (MB)
300	24,3	6.48
200	23,2	6,15
100	24,2	5,47
50	24,3	5,15

Tabulka 7.2: Tabulka statistiky modulu pro kontrolu úkolů, který kontroluje 1848 hlaviček při různě zvolené velikosti bloku adres, které se mají kontrolovat.

Statistika aktualizace na novou verzi generování dokumentace

Při aktualizaci se spouští všechny moduly kromě modulu pro kontrolu projektů. Výsledky statistiky jsou vidět v tabulce 7.3. Sloupec čas na hlavičku u projektů a skupin vyjadřuje čas, za který byla vytvořena nová hlavička. Čas, který zabere vyseparování dat ze staré hlavičky, není v tomto sloupci započten.

Modul	Počet URL	Doba běhu modulu (s)	Počet hlaviček	Čas na hlavičku (ms)	Max. paměť (MB)
Project Groups	1	0,12	1	100	1,48
Skupiny Projektů	10	2,35	10	25	1,69
Projekty	496	97,8	502	13	3,19
Celková Statistika	507	100,8	513	-	3,19

Tabulka 7.3: Tabulka statistiky aktualizace staré verze hlaviček na novou verzi.

Statistika běhu programu

Tabulka 7.4 ukazuje statistiku výkonu programu. Pro porovnání starý program, který kontroluje pouze skupiny projektů a projekty, běží na serveru skupiny KNOT skoro 6 minut a to není stará struktura hlaviček tak komplexní, jako ta nová.

Modul	Počet URL	Doba běhu modulu (s)	Počet hlaviček	Čas na hlavičku (ms)	Max. paměť (MB)
Project Groups	1	0,06	1	500	1,32
Skupiny Projektů	10	0,67	10	1	1,67
Projekty	496	31,65	502	0.009	3,12
Úkoly	455	25,2	1848	0.003	6,96
Celková Statistika	962	57,71	2361	-	6,96

Tabulka 7.4: Tabulka statistiky programu při normálním spuštění.

Kapitola 8

Závěr

Cílem této práce bylo vylepšit dosavadní způsob generování dokumentace z informačního systému KNOTIS, který byl chybový a nedostatečný. Po nastudování potřebných materiálů a pochopení problematiky jsem se rozhodl vytvořit úplně nový systém a program pro kontrolu a generování dokumentace. Tento nový systém funguje jako platforma, do které je možné přidávat funkcionalitu pomocí zásuvných modulů. Před implementací jsem také upravil soubory informačního systému KNOTIS, aby byly kompatibilní s novou verzí jazyka PHP, ve které jsem svůj program implementoval.

Po implementaci základu této platformy a pomocných modulů pro její fungování jsem vytvořil detailní návrh modulů pro tvorbu a kontrolu dokumentace. Tento návrh je tvořen diagramy aktivit, které zároveň působí jako dokumentace jejich funkčnosti. Daný způsob návrhu se velmi osvědčil při implementaci, kdy jsem mohl implementovat jednotlivé moduly podle těchto diagramů. Díky tomu tyto diagramy slouží i jako dokumentace implementace modulů a významným způsobem ulehčují orientaci v kódu a řešení případných chyb implementace.

Do svého návrhu jsem zapracoval opravy všech významných nedostatků minulé implementace, jako je třeba nedostatečná granularita nastavení generování hlaviček nebo nevyužití veškerých informací, které poskytuje informační systém KNOTIS. K těmto opravám jsem přidal i mnoho nových funkcí.

Nově se seznam skupin projektů negeneruje ručně, ale automaticky. Přidal jsem možnost umístění hlaviček skupin projektů a projektů na jedné stránce. Zjemnil jsem granularitu nastavení generování, kdy se nově může upravit generování jednotlivých hlaviček a ne jen vypínat celé moduly. Přidal jsem postup pro rozpoznání uživatelského textu v hlavičce, který na rozdíl od starého řešení umí rozeznat uživatelskou změnu od změny v databázi a umí vytvořit o této změně záznam. Tento nový postup nevytváří pouze jednotnou univerzální poznámku, ale poznámku podle druhu změny dat, například jestli uživatel přidal nebo změnil data, a specifikuje jaká data upravil. Také jsem navrhl a implementoval postup, který urychluje kontrolu hlaviček i běh celého programu. Dále jsem upravil strukturu hlaviček a přidal do nich další informace ze systému KNOTIS.

Výsledkem této práce je program v jazyce PHP. Tento program obsahuje moduly pro kontrolu skupin projektů a projektů, které kompletně nahrazují starý program a tvoří zá-

klad platformy pro kontrolu a generování dokumentace. Dále byl implementován modul pro seznam skupin projektů. Modul pro mazání stránek projektů řeší problém s nadbytečnými stránkami odstraněných projektů a demonstruje možnost propojení systému KNOTIS a platformy pro generování dokumentace. Modul pro kontrolu úkolů pak reprezentuje rozšíření této práce o funkcionalitu, která nebyla nikdy ve starém programu použita, a ukazuje možnosti snadného přidání další funkcionality do nové platformy.

Podařilo se mi splnit všechny části zadání této práce, přidat k nim funkcionalitu navíc a tím rozšířit některé body zadání. Výsledný program umí pracovat s nejnovějšími verzemi nástrojů a knihoven, obsahuje detailní dokumentaci pomocí diagramů, automaticky vygenerovanou dokumentaci metod a mnoho poznámek přímo v kódu. Podařilo se mi snížit dobu kontroly z 6 minut na 1,5 minuty a to program pracuje se složitější a sofistikovanější strukturou hlaviček. Program prošel důkladným testováním a chystá se jeho nasazení v ostrém provozu. To ukazuje, že návrh a implementace programu jsou kvalitní.

V blízké budoucnosti plánuji rozšířit program o další moduly, například modul pro kontrolu výkazů. Informační systém KNOTIS i platforma jsou pro tento modul již připraveny a uzpůsobeny. Ve vzdálenějším horizontu plánuji modul pro e-maily, který by byl schopen generovat dokumentaci z e-mailové korespondence mezi vedoucím a řešitelem.

Literatura

[1] Kaganer, P.; aj.: *MediaWiki Formátování*. [Online; navštíveno 07.05.2017].
URL https://www.mediawiki.org/wiki/Help:Formatting/cs

[2] Kolektiv autorů: *phpDocumentor analyzes your code*. [Online; navštíveno 10.05.2017].
URL https://www.phpdoc.org/

[3] Lopez, A.: *Learning PHP 7*. Packt Publishing, 2016, ISBN 1-78588-341-0.

[4] Megasaxon, C. R.: *PHP extension for the xxhash library*. [Online; navštíveno 01.05.2017].
URL https://github.com/Megasaxon/php-xxhash/tree/feature/php7

[5] Mehdi Achour, A. D., Friedhelm Betz; aj.: *Client URL Library*. [Online; navštíveno 07.05.2017].
URL http://php.net/manual/en/book.curl.php#book.curl

[6] Mehdi Achour, A. D., Friedhelm Betz; aj.: *Introduction to Php*. [Online; navštíveno 07.05.2017].
URL http://php.net/manual/en/introduction.php/

[7] Patrick Grässle, P. B., Henriette Baumann: *UML 2.0 in action : a project-based tutorial*. Packt Publishing, 2005, ISBN 1-904811-55-8.

[8] Prettyman, S.: *Learn PHP 7*. Apress, 2016, ISBN 978-1-4842-1729-0.

[9] Schwartz, B.: *MySQL profesionálně : optimalizace pro vysoký výkon*. Zoner Press, 2009, ISBN 978-80-7413-035-9.

[10] Tisza Gergő, P. B.; aj.: *MediaWiki hesla pro boty*. [Online; navštíveno 07.05.2017].
URL https://www.mediawiki.org/wiki/Manual:Bot_passwords

[11] Vavřínek, A.: *Automatické generování wiki stránek z e-mailů a IS*. Brno, 2011. Diplomová práce. Vysoké učení technické v brně, Fakulta informačních technologií, Ústav počítačové grafiky a multimédií. Vedoucí práce Ing. Jaroslav Dytrych. Dostupné z: https://www.vutbr.cz/www_base/zav_prace_soubor_verejne.php?file_id=117903.

[12] Yank, K.: *PHP & MySQL: Novice to Ninja, 5th Edition*. SitePoint, 2012, ISBN 978-0-9871530-8-1.

[13] Yuri Astrakhan, R. K.; aj.: *MediaWiki API*. [Online; navštíveno 07.05.2017].
URL https://www.mediawiki.org/wiki/API:Main_page

[14] Zendulka, J.: *Databázové systémy – 4 Relační model dat*. [Online; navštíveno 07.05.2017].
URL
http://www.fit.vutbr.cz/study/courses/DSI/public/pdf/nove/4_relmod.pdf

[15] Čapek, D.: *Databázový wrapper*. [Online; navštíveno 07.05.2017].
URL https://www.itnetwork.cz/php/mvc/objektovy-mvc-redakcni-system-v-php-pdo-crud-wrapped

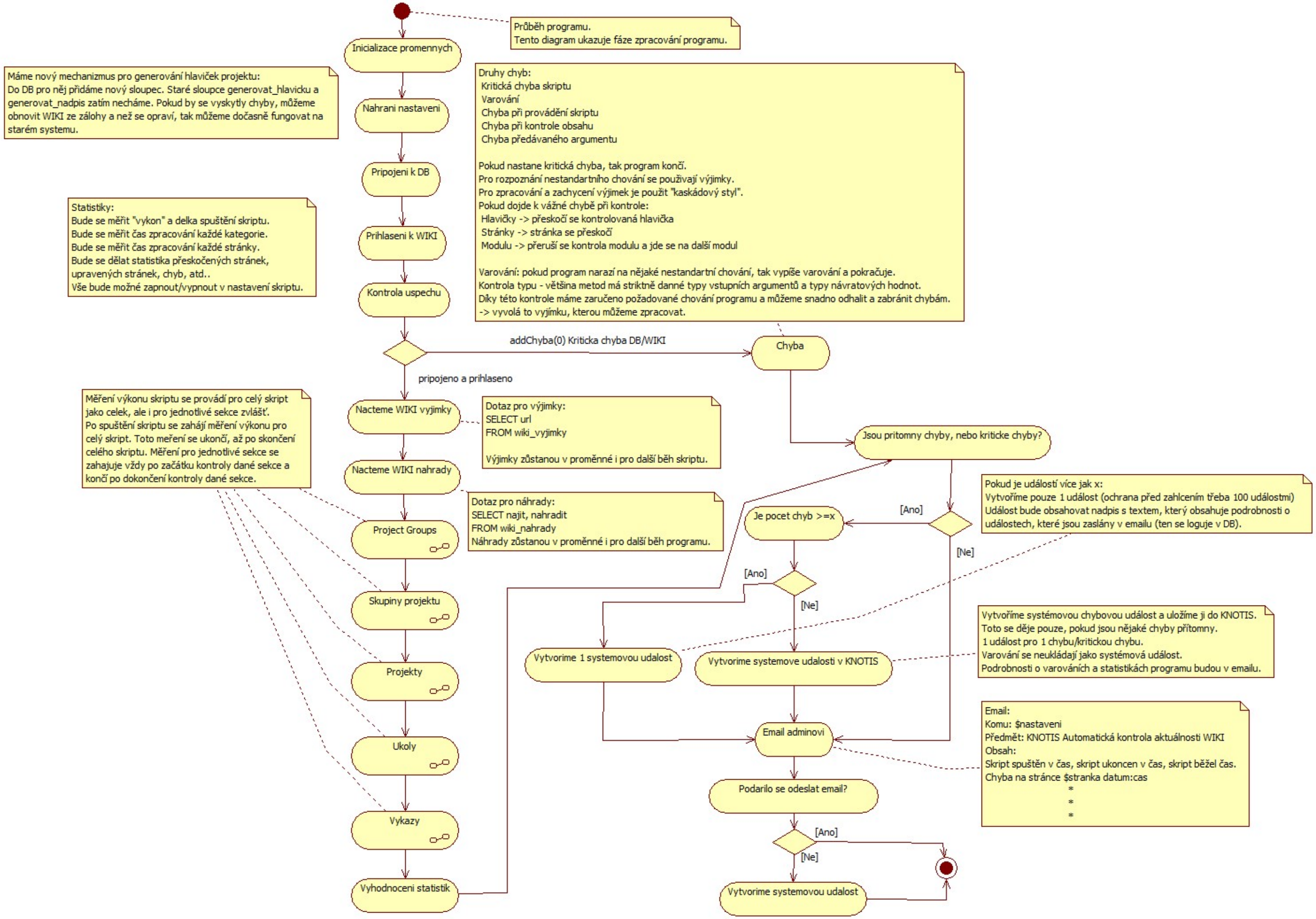
Přílohy

Příloha A

Diagramy aktivit

Zde jsou umístěny diagramy aktivit, které popisují průběh a chování programu.

A.1 Diagram průběhu programu


Průběh programu.
Tento diagram ukazuje fáze zpracování programu.
Inicializace promennych
Nahrani nastaveni
Pripojeni k DB
Prihlaseni k WIKI
Kontrola uspechu
Nacteme WIKI vyjimky
Nacteme WIKI nahrady
Project Groups
Skupiny projektu
Projekty
Ukoly
Vykazy
Vyhodnoceni statistik
Máme nový mechanizmus pro generování hlaviček projektu:
Do DB pro něj přidáme nový sloupec. Staré sloupce generovat_hlavicku a generovat_nadpis zatím necháme. Pokud by se vyskytly chyby, můžeme obnovit WIKI ze zálohy a než se opraví, tak můžeme dočasně fungovat na starém systemu.
Statistiky:
Bude se měřit "vykon" a delka spuštění skriptu.
Bude se měřit čas zpracování každé kategorie.
Bude se měřit čas zpracování každé stránky.
Bude se dělat statistika přeskočených stránek, upravených stránek, chyb, atd..
Vše bude možné zapnout/vypnout v nastavení skriptu.
Druhy chyb:
Kritická chyba skriptu
Varování
Chyba při provádění skriptu
Chyba při kontrole obsahu
Chyba předávaného argumentu
Pokud nastane kritická chyba, tak program končí.
Pro rozpoznání nestandartního chování se používají výjimky.
Pro zpracování a zachycení výjimek je použit "kaskádový styl".
Pokud dojde k vážné chybě při kontrole:
Hlavičky -> přeskočí se kontrolovaná hlavička
Stránky -> stránka se přeskočí
Modulu -> přeruší se kontrola modulu a jde se na další modul
Varování: pokud program narazí na nějaké nestandartní chování, tak vypíše varování a pokračuje.
Kontrola typu - většina metod má striktně danné typy vstupních argumentů a typy návratových hodnot.
Díky této kontrole máme zaručeno požadované chování programu a můžeme snadno odhalit a zabránit chybám.
-> vyvolá to vyjímku, kterou můžeme zpracovat.
addChyba(0) Kriticka chyba DB/WIKI
Chyba
pripojeno a prihlaseno
Měření výkonu skriptu se provádí pro celý skript jako celek, ale i pro jednotlivé sekce zvlášť.
Po spuštění skriptu se zahájí měření výkonu pro celý skript. Toto meření se ukončí, až po skončení celého skriptu. Měření pro jednotlivé sekce se zahajuje vždy po začátku kontroly dané sekce a končí po dokončení kontroly dané sekce.
Dotaz pro výjimky:
SELECT url
FROM wiki_vyjimky
Výjimky zůstanou v proměnné i pro další běh skriptu.
Dotaz pro náhrady:
SELECT najit, nahradit
FROM wiki_nahrady
Náhrady zůstanou v proměnné i pro další běh programu.
Jsou pritomny chyby, nebo kriticke chyby?
[Ano]
[Ne]
Je pocet chyb >=x
[Ano]
[Ne]
Pokud je událostí více jak x:
Vytvoříme pouze 1 událost (ochrana před zahlcením třeba 100 událostmi)
Událost bude obsahovat nadpis s textem, který obsahuje podrobnosti o událostech, které jsou zaslány v emailu (ten se loguje v DB).
Vytvorime 1 systemovou udalost
Vytvorime systemove udalosti v KNOTIS
Vytvoříme systémovou chybovou událost a uložíme ji do KNOTIS.
Toto se děje pouze, pokud jsou nějaké chyby přítomny.
1 událost pro 1 chybu/kritickou chybu.
Varování se neukládají jako systémová událost.
Podrobnosti o varováních a statistikách programu budou v emailu.
Email adminovi
Email:
Komu: $nastaveni
Předmět: KNOTIS Automatická kontrola aktuálnosti WIKI
Obsah:
Skript spuštěn v čas, skript ukoncen v čas, skript běžel čas.
Chyba na stránce $stranka datum:cas
*
*
*
Podarilo se odeslat email?
[Ano]
[Ne]
Vytvorime systemovou udalost

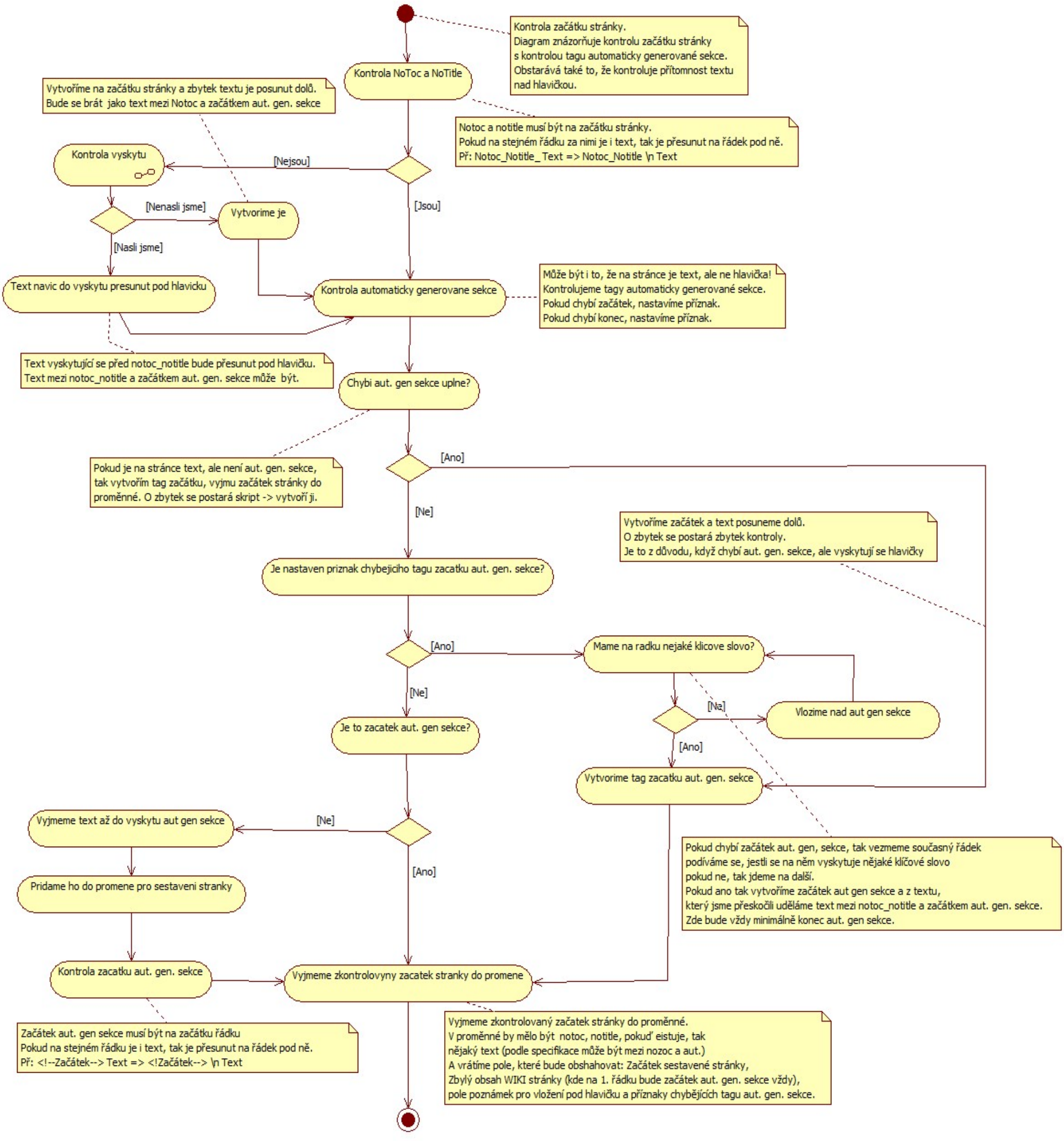
A.2 Subdiagramy použité v diagramech modulů

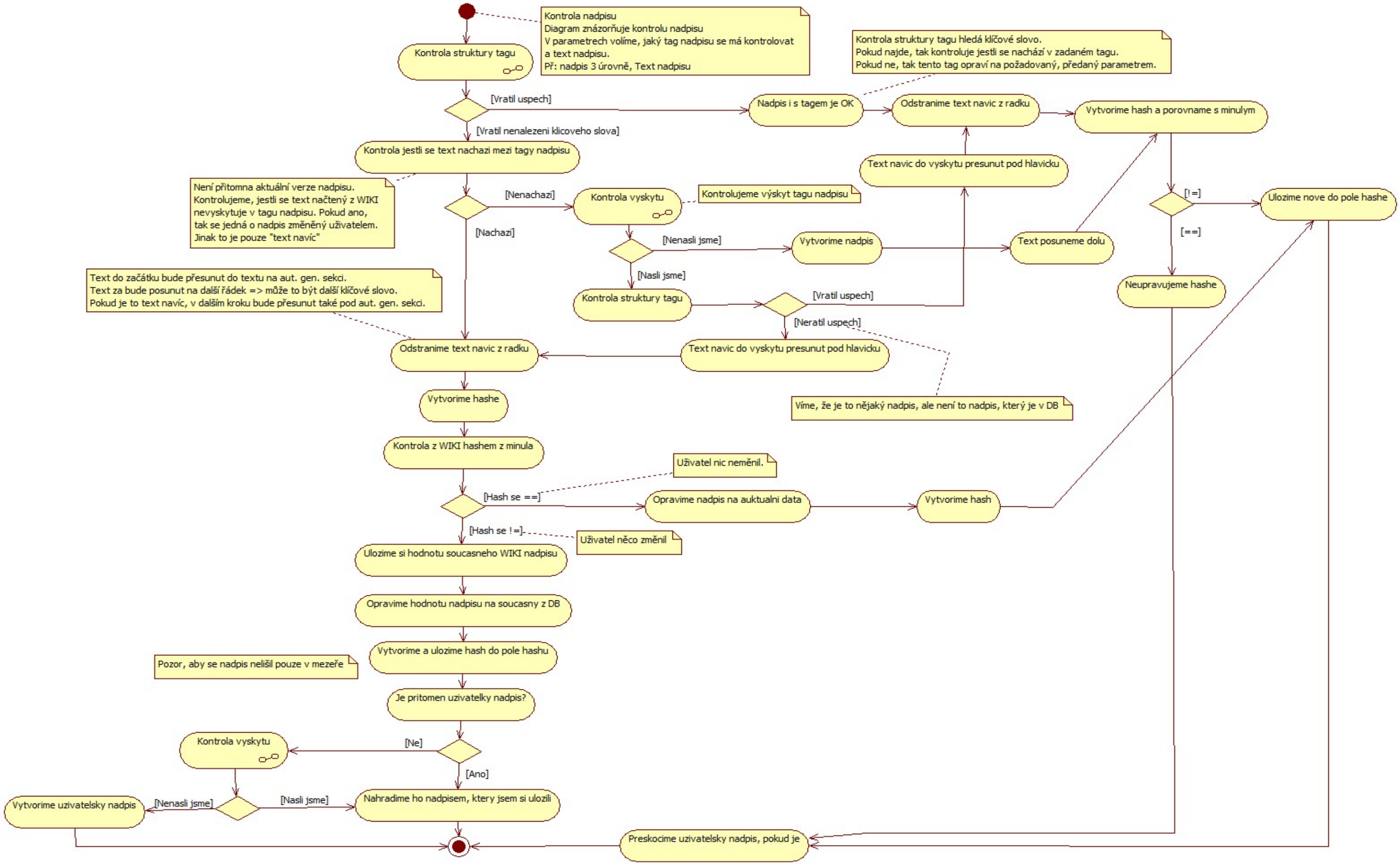
Kontrola začátku stránky

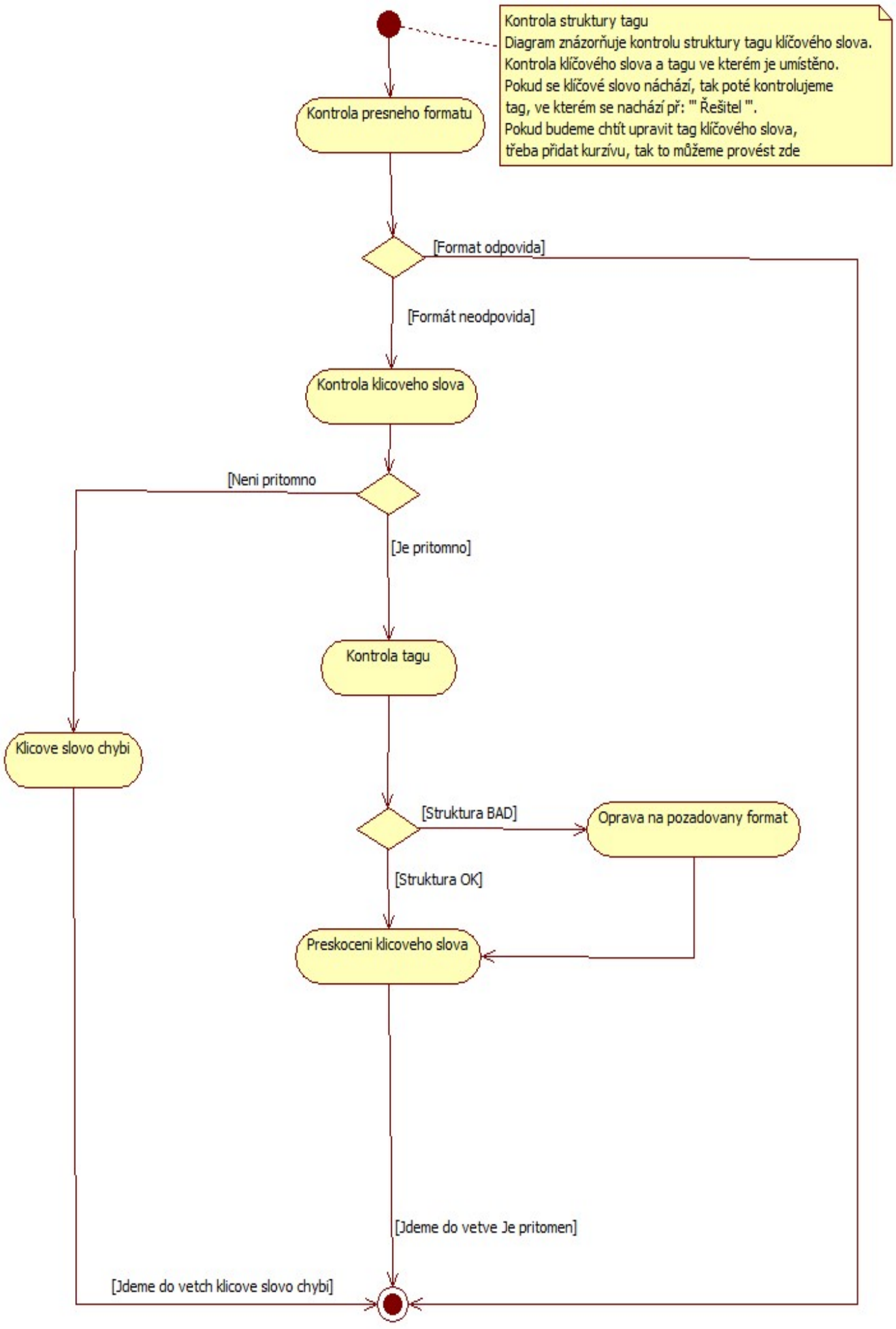
Kontrola nadpisu

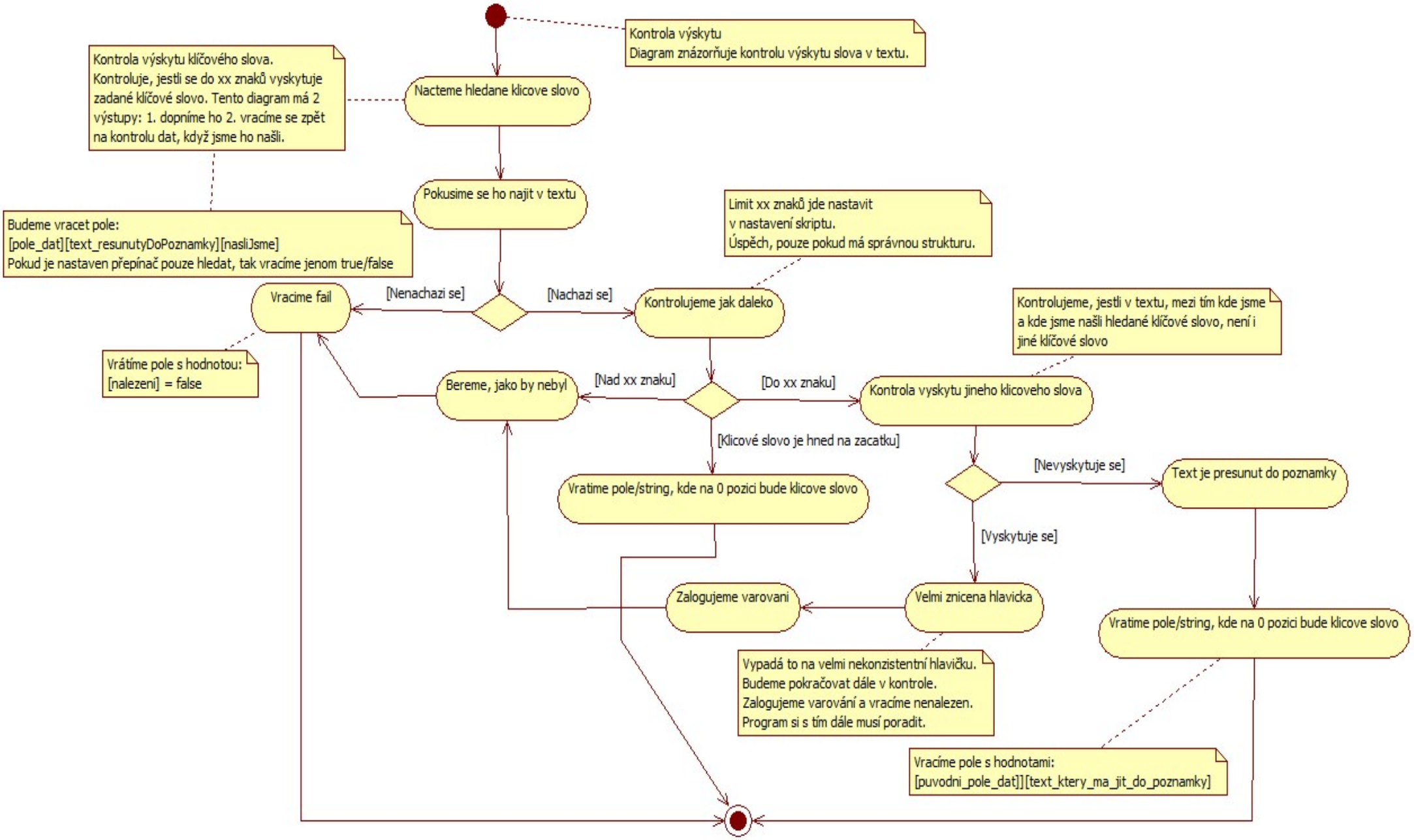
Kontrola struktury tagu

Kontrola výskytu


Kontrola začátku stránky.
Diagram znázorňuje kontrolu začátku stránky
s kontrolou tagu automaticky generované sekce.
Obstarává také to, že kontroluje přítomnost textu
nad hlavičkou.
Kontrola NoToc a NoTitle
Notoc a notitle musí být na začátku stránky.
Pokud na stejném řádku za nimi je i text, tak je přesunut na řádek pod ně.
Př: Notoc_Notitle_ Text => Notoc_Notitle \n Text
Vytvoříme na začátku stránky a zbytek textu je posunut dolů.
Bude se brát jako text mezi Notoc a začátkem aut. gen. sekce
Kontrola vyskytu
[Nejsou]
[Nenasli jsme]
Vytvorime je
[Nasli jsme]
[Jsou]
Text navic do vyskytu presunut pod hlavicku
Kontrola automaticky generovane sekce
Může být i to, že na stránce je text, ale ne hlavička!
Kontrolujeme tagy automaticky generované sekce.
Pokud chybí začátek, nastavíme příznak.
Pokud chybí konec, nastavíme příznak.
Text vyskytující se před notoc_notitle bude přesunut pod hlavičku.
Text mezi notoc_notitle a začátkem aut. gen. sekce může být.
Chybi aut. gen sekce uplne?
Pokud je na stránce text, ale není aut. gen. sekce,
tak vytvořím tag začátku, vyjmu začátek stránky do
proměnné. O zbytek se postará skript -> vytvoří ji.
[Ano]
[Ne]
Vytvoříme začátek a text posuneme dolů.
O zbytek se postará zbytek kontroly.
Je to z důvodu, když chybí aut. gen. sekce, ale vyskytují se hlavičky
Je nastaven priznak chybejiciho tagu zacatku aut. gen. sekce?
[Ano]
Mame na radku nejaké klicove slovo?
[Ne]
Je to zacatek aut. gen sekce?
[Ne]
Vlozime nad aut gen sekce
[Ano]
Vytvorime tag zacatku aut. gen. sekce
Vyjmeme text až do vyskytu aut gen sekce
[Ne]
[Ano]
Pridame ho do promene pro sestaveni stranky
Pokud chybí začátek aut. gen, sekce, tak vezmeme současný řádek
podíváme se, jestli se na něm vyskytuje nějaké klíčové slovo
pokud ne, tak jdeme na další.
Pokud ano tak vytvoříme začátek aut gen sekce a z textu,
který jsme přeskočili uděláme text mezi notoc_notitle a začátkem aut. gen. sekce.
Zde bude vždy minimálně konec aut. gen sekce.
Kontrola zacatku aut. gen. sekce
Vyjmeme zkontrolovyny zacatek stranky do promene
Začátek aut. gen sekce musí být na začátku řádku
Pokud na stejném řádku je i text, tak je přesunut na řádek pod ně.
Př: <!--Začátek--> Text => <!Začátek--> \n Text
Vyjmeme zkontrolovaný začatek stránky do proměnné.
V proměnné by mělo být notoc, notitle, pokuď eistuje, tak
nějaký text (podle specifikace může být mezi nozoc a aut.)
A vrátíme pole, které bude obshahovat: Začátek sestavené stránky,
Zbylý obsah WIKI stránky (kde na 1. řádku bude začátek aut. gen. sekce vždy),
pole poznámek pro vložení pod hlavičku a příznaky chybějících tagu aut. gen. sekce.


Kontrola nadpisu
Diagram znázorňuje kontrolu nadpisu
V parametrech volíme, jaký tag nadpisu se má kontrolovat
a text nadpisu.
Př: nadpis 3 úrovně, Text nadpisu
Kontrola struktury tagu
[Vratil uspech]
[Vratil nenalezeni klicoveho slova]
Kontrola struktury tagu hledá klíčové slovo.
Pokud najde, tak kontroluje jestli se nachází v zadaném tagu.
Pokud ne, tak tento tag opraví na požadovaný, předaný parametrem.
Nadpis i s tagem je OK
Odstranime text navic z radku
Vytvorime hash a porovname s minulym
Kontrola jestli se text nachazi mezi tagy nadpisu
Není přitomna aktuální verze nadpisu.
Kontrolujeme, jestli se text načtený z WIKI
nevyskytuje v tagu nadpisu. Pokud ano,
tak se jedná o nadpis změněný uživatelem.
Jinak to je pouze "text navíc"
[Nenachazi]
Kontrola vyskytu
Kontrolujeme výskyt tagu nadpisu
[Nachazi]
[Nenasli jsme]
Vytvorime nadpis
Text navic do vyskytu presunut pod hlavicku
Text posuneme dolu
[!=]
Ulozime nove do pole hashe
[==]
Neupravujeme hashe
[Nasli jsme]
Kontrola struktury tagu
[Vratil uspech]
[Neratil uspech]
Text do začátku bude přesunut do textu na aut. gen. sekci.
Text za bude posunut na další řádek => může to být další klíčové slovo.
Pokud je to text navíc, v dalším kroku bude přesunut také pod aut. gen. sekci.
Odstranime text navic z radku
Text navic do vyskytu presunut pod hlavicku
Víme, že je to nějaký nadpis, ale není to nadpis, který je v DB
Vytvorime hashe
Kontrola z WIKI hashem z minula
Uživatel nic neměnil.
[Hash se ==]
Opravime nadpis na auktualni data
Vytvorime hash
[Hash se !=]
Uživatel něco změnil
Ulozime si hodnotu soucasneho WIKI nadpisu
Opravime hodnotu nadpisu na soucasny z DB
Vytvorime a ulozime hash do pole hashu
Pozor, aby se nadpis nelišil pouze v mezeře
Je pritomen uzivatelky nadpis?
Kontrola vyskytu
[Ne]
[Ano]
Vytvorime uzivatelsky nadpis
[Nenasli jsme]
[Nasli jsme]
Nahradime ho nadpisem, ktery jsem si ulozili
Preskocime uzivatelsky nadpis, pokud je


Kontrola struktury tagu
Diagram znázorňuje kontrolu struktury tagu klíčového slova.
Kontrola klíčového slova a tagu ve kterém je umístěno.
Pokud se klíčové slovo náchází, tak poté kontrolujeme
tag, ve kterém se nachází př: '" Řešitel '".
Pokud budeme chtít upravit tag klíčového slova,
třeba přidat kurzívu, tak to můžeme provést zde
Kontrola presneho formatu
[Format odpovida]
[Formát neodpovida]
Kontrola klicoveho slova
[Neni pritomno
[Je pritomno]
Kontrola tagu
Klicove slovo chybi
[Struktura BAD]
Oprava na pozadovany format
[Struktura OK]
Preskoceni klicoveho slova
[Jdeme do vetve Je pritomen]
[Jdeme do vetch klicove slovo chybi]


Kontrola výskytu
Diagram znázorňuje kontrolu výskytu slova v textu.
Kontrola výskytu klíčového slova.
Kontroluje, jestli se do xx znaků vyskytuje zadané klíčové slovo. Tento diagram má 2 výstupy: 1. dopníme ho 2. vracíme se zpět na kontrolu dat, když jsme ho našli.
Nacteme hledane klicove slovo
Pokusime se ho najit v textu
Budeme vracet pole:
[pole_dat][text_resunutyDoPoznamky][nasliJsme]
Pokud je nastaven přepínač pouze hledat, tak vracíme jenom true/false
Limit xx znaků jde nastavit
v nastavení skriptu.
Úspěch, pouze pokud má správnou strukturu.
Vracime fail
[Nenachazi se]
[Nachazi se]
Kontrolujeme jak daleko
Kontrolujeme, jestli v textu, mezi tím kde jsme a kde jsme našli hledané klíčové slovo, není i jiné klíčové slovo
Vrátíme pole s hodnotou:
[nalezeni] = false
Bereme, jako by nebyl
[Nad xx znaku]
[Do xx znaku]
Kontrola vyskytu jineho klicoveho slova
[Klicové slovo je hned na zacatku]
[Nevyskytuje se]
Text je presunut do poznamky
Vratime pole/string, kde na 0 pozici bude klicove slovo
[Vyskytuje se]
Zalogujeme varovani
Velmi znicena hlavicka
Vratime pole/string, kde na 0 pozici bude klicove slovo
Vypadá to na velmi nekonzistentní hlavičku.
Budeme pokračovat dále v kontrole.
Zalogujeme varování a vracíme nenalezen.
Program si s tím dále musí poradit.
Vracíme pole s hodnotami:
[puvodni_pole_dat]][text_ktery_ma_jit_do_poznamky]

A.3 Diagramy pro zpracování Project Groups

Diagram modulu Project Groups

Diagram vytvoření stránky Project Groups

Struktura Project groups:
__NOTITLE__NOTOC__
<!--Zacatek automaticky generovane sekce-->
(h1)Skupiny projektu
: Uživatelsky nadpis: Nadpis (pouze, kdyz uživatel změní nadpis ručne na svůj, neuchováváme historii nadpisu, pouze 1 poslední)
*<!--skupina_projektu:ID1-->[[URL | Nazev(zkratka)]]
: Poznámka... (možné poznámky jsou dole, " : " je identifikace poznámky)
*<!--skupina_projektu:ID2-->[[URL | Nazev(zkratka)]]
*
:Skupiny projektu, které nejsou v KNOTIS DB
*
----<!--Konec automaticky generovane sekce-->
:Nekonzistentní skupiny projektu
*
Možné typy poznámek:
Skupiny projektů, které nejsou v DB
uživatelský odkaz na skupinu: URL
uživatelské jméno skupiny: Jmeno
uživatelská zkratka skupiny: Zkratka
uživatelský text ke skupině projektů: text

Pokud splňuje 1+ kategorii poznámky, tak se vypíše:
Skupina projektů upravená uživatelem: [skupina projektu]

Struktura DB tabulky project_groups_heshs:
<<PK>>skupina_projektu_id
db_hash - skládá se z id, URL, názvu a zkratky
wiki_hash skládá se z toho, co je na řádku ve wiki (i s tagy)

Speciální záznam:
<<PK>>skupina_id = 0 - je to hash nadpisu stránky

Project groups
Tento diagram zobrazuje kontrolu stránky project groups.
URL stránky bude možné nastavit v KNOTIS v sekci údržba.
Pokud nebude URL zadané, tak se stránka bude generovat, na předdefinované adrese v nastavení.
Na této strance může být uživatelský obsah.

Načte se ID, zkratka a URL, řazeno v dotazu podle názvu a zkratky podle abecedy.
Data se načtou do pole DB dat.

Hledaní WIKI stránky:
Hledá se podle titulu.
Když máme stránku xxx/index.php/dict -> hledáme dict
Když máme xxx/index.php/neco/dict -> hledame neco/dict

Mechanizmus:
podle nastavení skriptu ořízneme URL
když jsou short tags, tak po wiki/
když nejsou short tags tak po index.php/

Sestavíme do proměnné/pole obsah stránky z DB dat.
Do tabulky project group hash přidáme hashe jednotlivých skupin.
Získame editační token.
Editujeme stránku.

Načteme data z WIKI:
Načteme poslední revizi stránky.

Kontrolujeme tag negenerovat před začátkem aut. gen. sekce
V nastavení můžeme nastavit, do kolika znaků se má hledat.

Něco se na stránce změnilo, budeme ji kontrolovat.
Pokud byla aut. sekce ok, tak pracujeme jenom s ní
(je vyjmuta), pokud nebyla pracujeme s celou stránou.
Načteme z DB hashe poslední kontroly.
Pokud se nám nepodaří načíst hashe (dotaz vrátí MySQL error),
tak se jedná o chybu a kontrolu stránky přeskakujeme.
Pokud jsou hodnoty NULL, tak pokračujeme (ještě skript tuto
stránku nekontroloval a nebyly vytvořeny).

Notitle a notoc se vytvářejí na úplném začátku,
před tagem automaticky generované sekce.
Tag <negenerovat> může být pouze na úplném
začátku stránky.

Nadpis pro tuto stránku je uložen v DB tabulka ostatní.
Dá se měnit v KNOTIS IS.
Pokud nadpis na stránce chybí, je doplněn a text se posune dolů.

Přeskakujeme prázdné řádky a řádky začínající :poznámky.

Musíme zkontrolovat, jestli se tam ta samá
poznámka už nevyskytuje - stejný text poznámky.

Pokud za tagem s ID není platná struktura, podle preg matche -> špatně.
Pokud se ID nevyskytuje v datech z DB, je to skupina projektů, která již není v DB.
Pokud se vyskytuje jiný text, než sem patří, je přemístěn do pole poznámek
jako uživatelský text.
Struktura: Pozor na vnitřní a vnější odkaz
TAG ODKAZ
*<!--skupina_projektu:ID1-->[[URL | Nazev(zkratka)]]

Skupina je i s případnými daty navíc přemístěna
do Projekty, které již nejsou v DB KNOTIS.

Zkontrolujeme, jestli v datech není
platná skupina projektů. Jestli ne, tak
přesuneme data do poznámky pod skupinu.

Pokud budou v poli více jak 2 poznámky,
tak už nekontrolujeme a rovnou dáme do poznámky
Skupina projektu upravená uživatelem : data řádku.

Wiki hashe se !=
i když se != DB hashe bereme,
že uživatel něco změnil což
má největší prioritu

DB hashe se !=, ale
WIKI hashe se ==
Uživatel nic neupravil.

Pokud v poli dat z Db zbyla data, pro ne vytvorime skupiny

Má to ID, skript už to minimálně 1 musel projet.
Pokud se ID nachází v DB datech, tak opravíme a
uděláme poznámku na dalším řádku. Pokud se ID
nenachází v DB datech, tak projekt přesuneme do
sekce nekonzistentní skupiny za tag konce aut. gen. sekce.

Kontrolujeme výskyt odkazu na skupinu projektů
s tagem s id dále v textu podle preg Matche.
Musím přidat do klíčových slov slovo pro nekonzistentni...

Dále už nemáme platná data, tak ukončíme cyklus

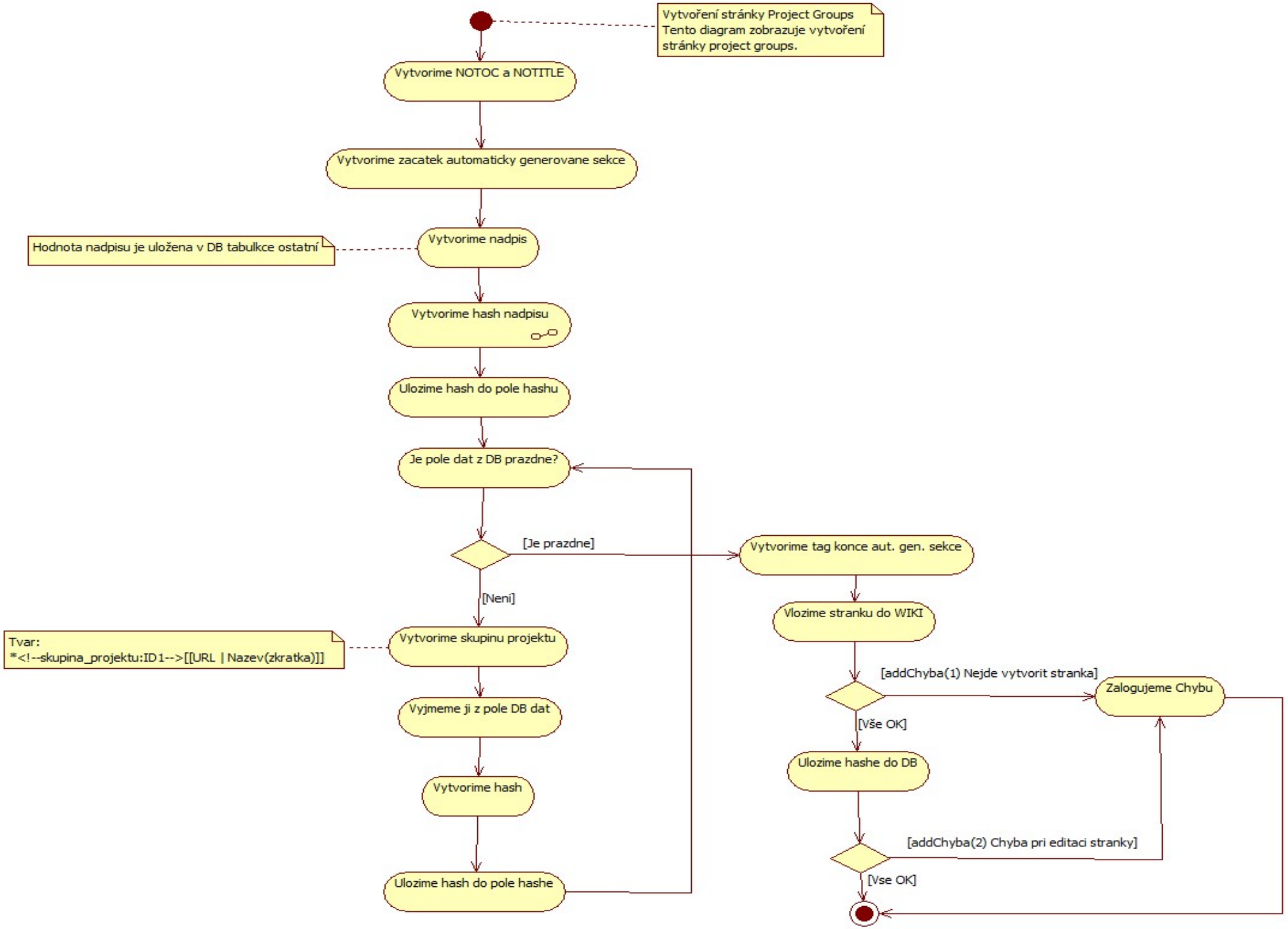
Pokud nepřidáme do poznámky
Skupina projektu upravená uživatelem : data řádku.

Pokud text za aut. gen. sekci není empty, přidat varování.
Také přidat varování, pokud chybí konec aut. gen. sekce.

Dát si pozor, ať nevkládáme hashe, které již jsou v DB zbytečně.
Také si dát pozor na to, jestli vkládáme/editujeme hash

Pokud není proměnná s textem navíc prázdná,
tak přidáme do pole i položku s poznámkou.

Project groups
addChyba(0) Data nenactena
Kriticka chyba
Data nactena
Nalezeni stranky na WIKI
Stranka existuje
Stranka neexistuje
Editace existujici stranky
Vytvareni nove stranky
Nacteni dat z WIKI
Vytvoreni stranky project groups
Je stranka prazdna?
[Ano]
[Ne]
Kontrola tagu negenerovat
[Je pritomen]
[Neni pritomen]
Nacteni hashe posledni kontroly
[Hashe se nenactely]
Chyba skriptu
[Heshe se nactely]
Nacteme podrobna data o skupinách z DB
[Data se nanactela]
[Data se nactela]
Zalogujeme chybu a přeskakujeme kontrolu Project groups.
Kontrola zacatku stranky
Kontrola nadpisu
Mame dalsi radky ke kontrole?
Pridáme do pole k posledni zkotrolovane skupine
Preskocime
[Ne]
[Ano]
Je to nadpis skupin, ktere nejsou v DB?
Co je na radku?
[Poznamka]
[Prazdny radek]
Preskocime
[Text]
Kontrola struktury skupiny
[Struktura neni OK]
[Struktura je OK]
Ma to tag s id?
Kontrola vyskytu ID v DB
[Nevyskytuje se]
[Vyskytuje se]
Skupina projektu navic
Kontrola dat navic
Presunuti do poznamek
[Jsou pritomna]
[Nejsou pritomna]
Vytvorime WIKI a Db hash
Porovname se starymi
[WIKI hashe !=]
[Oba se ==]
[DB hashe se !=]
Kontrola odkazu
Opravime odkaz
Pridame poznamku do pole
[Spatny odkaz]
[Spravny odkaz]
Opravime data na aktualni
Vytvorime a ulozime hashe do pole hashu
Kontrola jmena
Opravime jmeno
Pridame poznamku do pole
[Spatne jmeno]
[Spravne jmeno]
Kontrola zkratky
Opravime zkratku
Pridame poznamku do pole
[Spatna zkratka]
[Spravna zkratka]
Kontrola skupiny dokoncena
Vytvorime hash a ulozime ho do pole hashu
Vse je aktualni
Pridame skupinu do pole skupin
Z DB dat odstranime polozku s timto ID
Nachazi se ID v DB datech?
[Ano]
[Ne]
Presuneme do nekonzistentnich skupin
Pridame varovani
Opravime
Vytvorime a ulozime hash do pole
Id vyjmeme z DB dat
Je na radku pritomny odkaz?
[Ano]
[Ne]
Presuneme do nekonzistentnich skupin
Zkusime pomoci udaju v odkazu najit skupinu v DB
[Nenachazi se]
[Nachazi se]
Vytvorime skupinu projektu
Chybi konec aut. gen. sekce na strance?
[Ano]
[Ne]
Kontrola vyskytu
[Nevyskytuje se]
Ukoncujeme cyklus
Presuneme radky do dat za aut. gen. sekci
[Vyskytuje se]
A chybí konec aut. ge. sekce
Pokud v poli dat zbyla data vlozime je do promenne pro zbytek stranky
Seskupime pole skupin podle nazvu
Pridame skupiny projektu do sestavene stranky
Seskupime skupiny projektu navic
Pridame napis skupin navic do stranky
Pridame skupiny projektu navic do sestavene stranky
Pridame konec aut. gen sekce
Pridame sekci s nekonzistentnimi projekty
Pridame data za aut. gen. sekci do zbytku stranky
Pridame data v promenne zbytek stranky
Kontrola nahrad v datech pro zbytek stranky
Ulozime pole hashu do DB
Pomoci API editujeme


Vytvoření stránky Project Groups
Tento diagram zobrazuje vytvoření stránky project groups.
Vytvorime NOTOC a NOTITLE
Vytvorime zacatek automaticky generovane sekce
Vytvorime nadpis
Hodnota nadpisu je uložena v DB tabulkce ostatní
Vytvorime hash nadpisu
Ulozime hash do pole hashu
Je pole dat z DB prazdne?
[Je prazdne]
[Není]
Vytvorime tag konce aut. gen. sekce
Vlozime stranku do WIKI
[addChyba(1) Nejde vytvorit stranka]
Zalogujeme Chybu
[Vše OK]
Ulozime hashe do DB
[addChyba(2) Chyba pri editaci stranky]
[Vse OK]
Tvar:
*<!--skupina_projektu:ID1-->[[URL | Nazev(zkratka)]]
Vytvorime skupinu projektu
Vyjmeme ji z pole DB dat
Vytvorime hash
Ulozime hash do pole hashe

A.4 Diagramy pro zpracování Kontroly skupin projektů

Diagram modulu Skupiny projektu

Diagram vytvoření stránky skupiny projektů

Diagram pro kontrolu vedoucího

Diagram pro kontrolu Redmine

Diagram pro kontrolu externí URL

Diagram pro kontrolu prostředků

Diagram pro kontrolu projektů skupin projektů

Struktura stránky skupiny projektů:
<!--NEGENEROVAT-->
__notitle____notoc__
<!-- Zacatek automaticky generovane sekce -->
<!--skupina_projektu:ID-->
(h1)Nazev(zkratka)
:Uživatelský nadpis: xxx (pouze pokud uživatel změnil název)
Vedouci skupiny:<!--vedouci:ID-->tit Jmeno
:Vedoucí doplnění Uzivatelem: tit Jmeno
:Předchozí vedoucí: <!--predchozi_vedouci:ID-->tit Jmeno
:Předchozí vedoucí: <!--predchozi_vedouci:ID-->tit Jmeno
*
*
Redmine: URl
Externi URL: URL pokud neni nepsat

Prostředky skupiny:

nazev	cesta	opravneni	patri k projektu	server
x	x	x	x	x
x	x	zadne	x	x

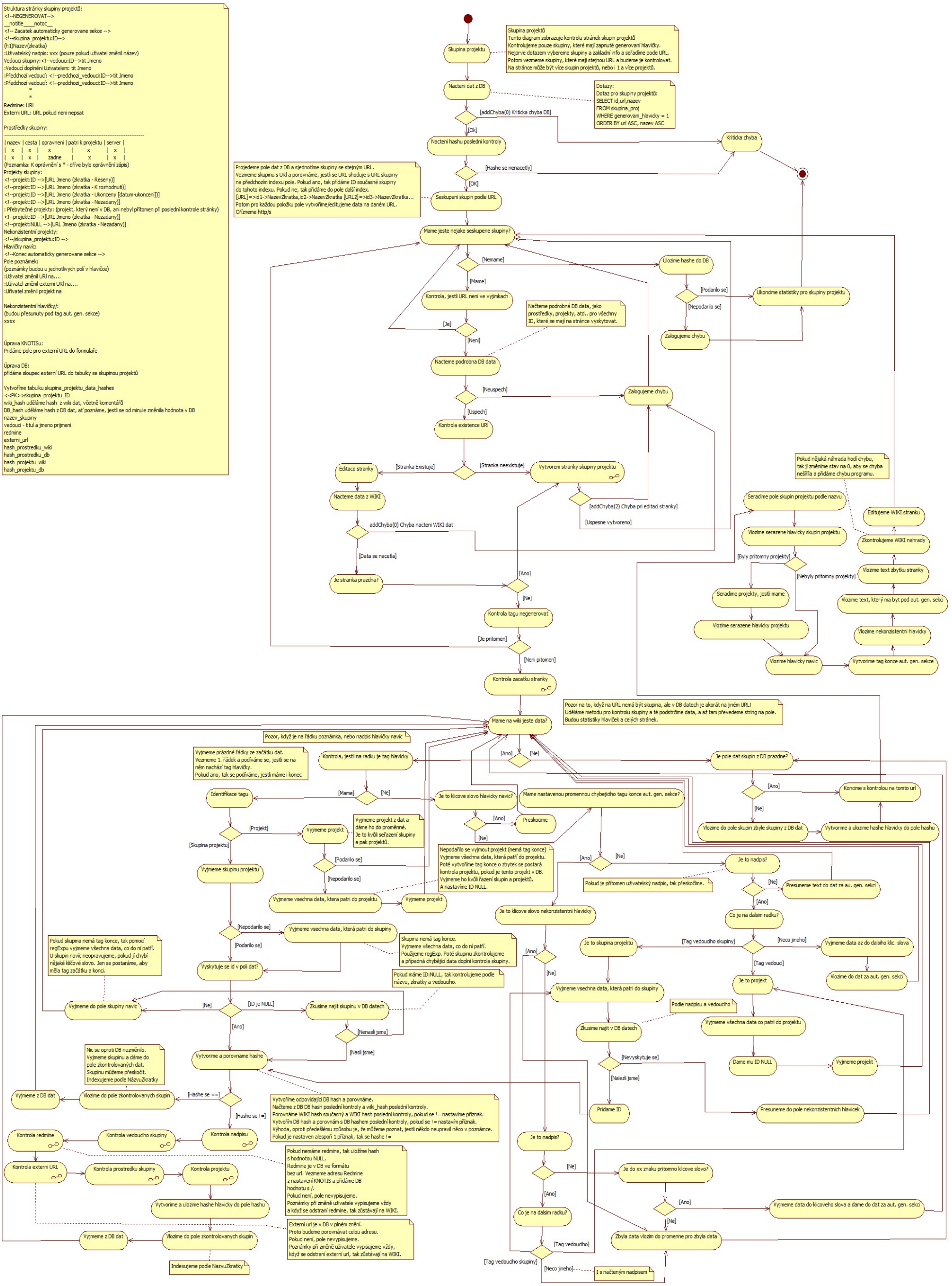
(Poznamka: K oprávnění s * - dříve bylo oprávnění zápis)
Projekty skupiny:
<!--projekt:ID -->[URL Jmeno (zkratka - Reseny)]
<!--projekt:ID -->[URL Jmeno (zkratka - K rozhodnuti)]
<!--projekt:ID -->[URL Jmeno (zkratka - Ukonceny [datum-ukonceni])]
<!--projekt:ID -->[URL Jmeno (zkratka - Nezadany)]
: Přebytečné projekty: (projekt, který není v DB, ani nebyl přítomen při poslední kontrole stránky)
<!--projekt:ID -->[URL Jmeno (zkratka - Nezadany)]
<!--projekt:NULL -->[URL Jmeno (zkratka - Nezadany)]
Nekonzistentní projekty:
<!--/skupina_projektu:ID -->
Hlavičky navíc:
<!--Konec automaticky generovane sekce -->
Pole poznámek:
(poznámky budou u jednotlivych polí v hlavičce)
:Uživatel změnil URl na....
:Uživatel změnil externi URl na....
:Uřivatel změnil projekt na

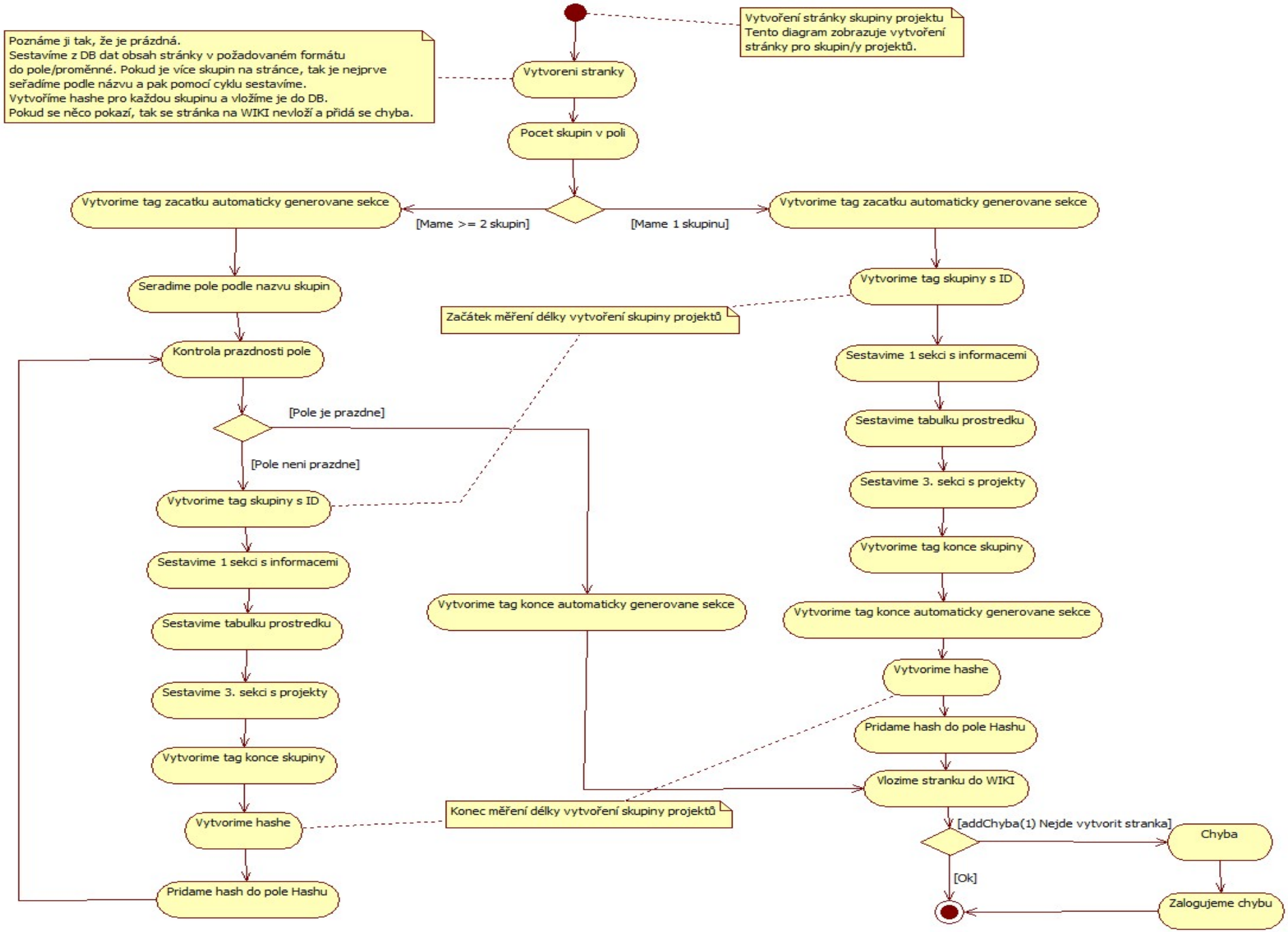
Nekonzistentní hlavičky/:
(budou přesunuty pod tag aut. gen. sekce)
xxxx

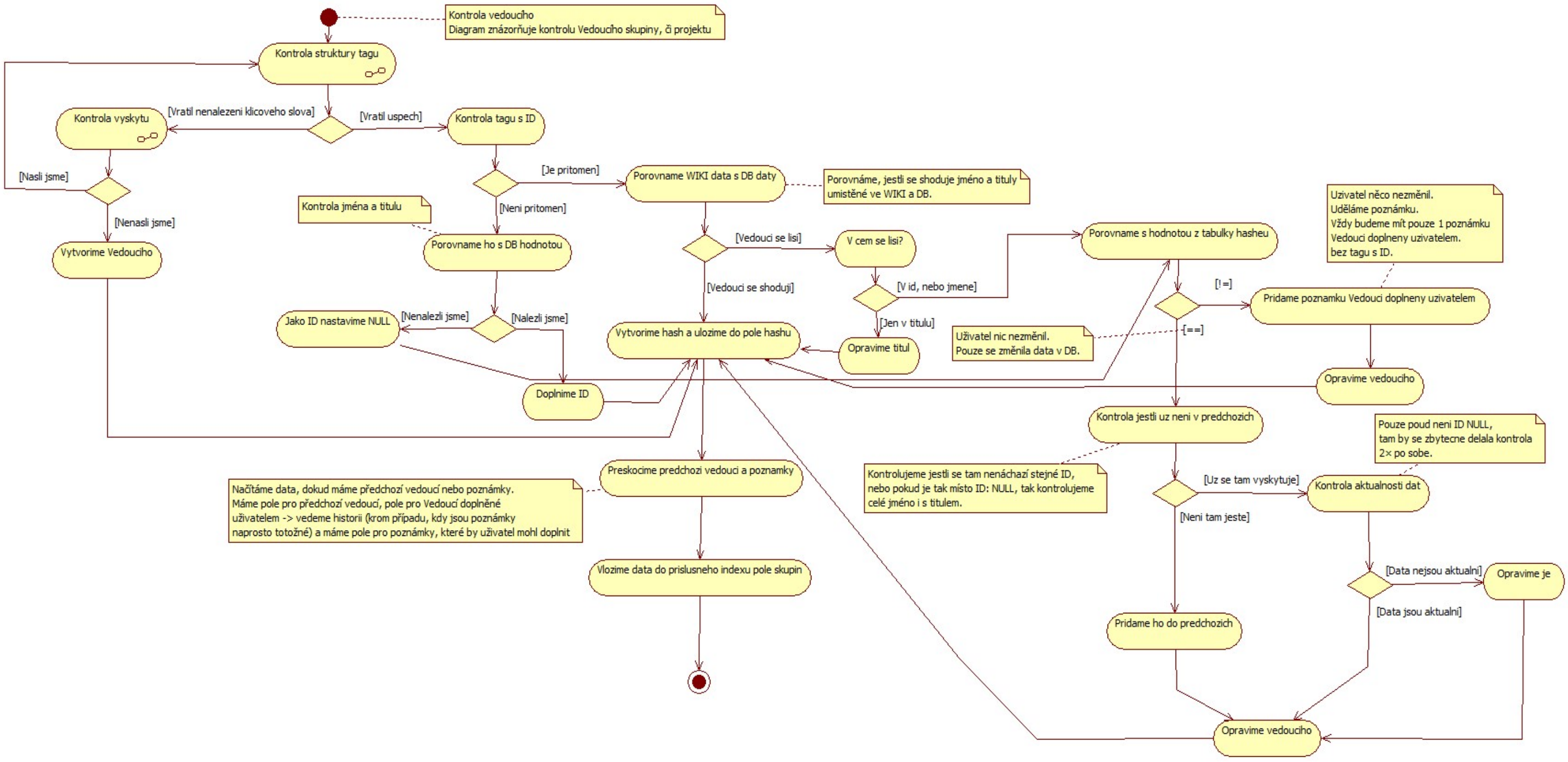
Úprava KNOTISu:
Pridáme pole pro externí URL do formuláře

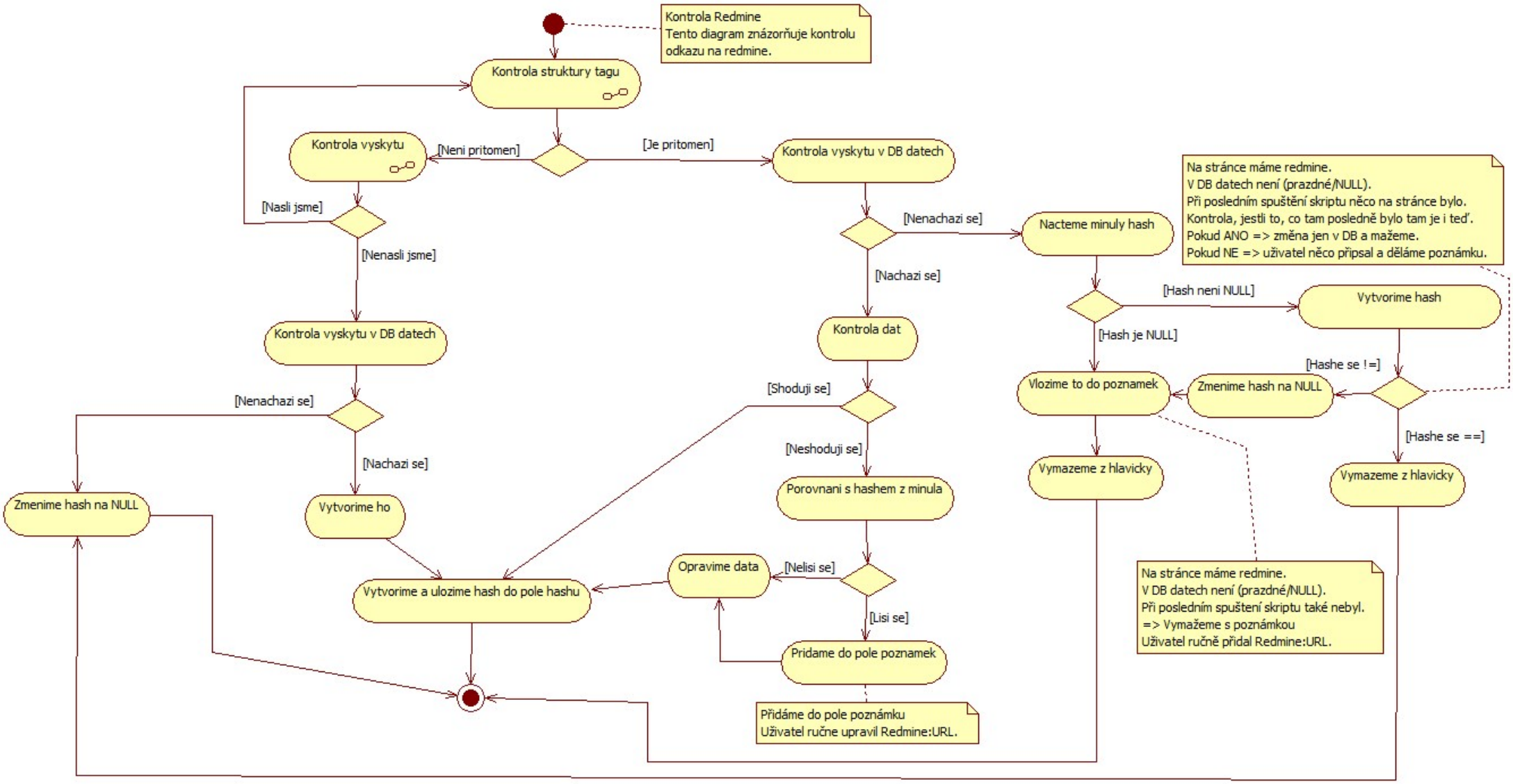
Úprava DB:
přidáme sloupec externí URL do tabulky se skupinou projektů

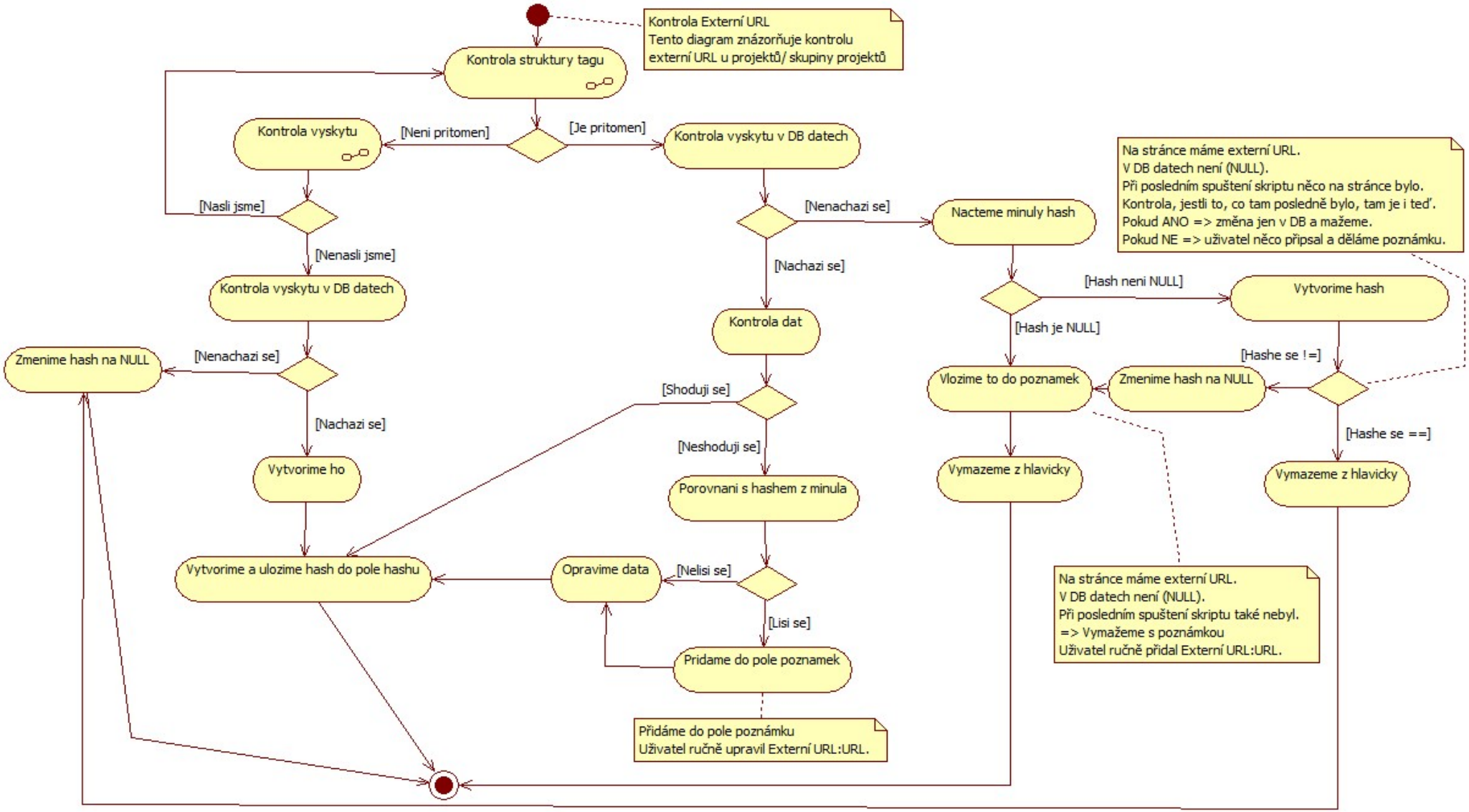
Vytvoříme tabulku skupina_projektu_data_hashes
<<PK>>skupina_projektu_ID
wiki_hash uděláme hash z wiki dat, včetně komentářů
DB_hash uděláme hash z DB dat, ať poznáme, jestli se od minule změnila hodnota v DB
nazev_skupiny
vedouci - titul a jmeno prijmeni
redmine
externi_url
hash_prostredku_wiki
hash_prostredku_db
hash_projektu_wiki
hash_projektu_db

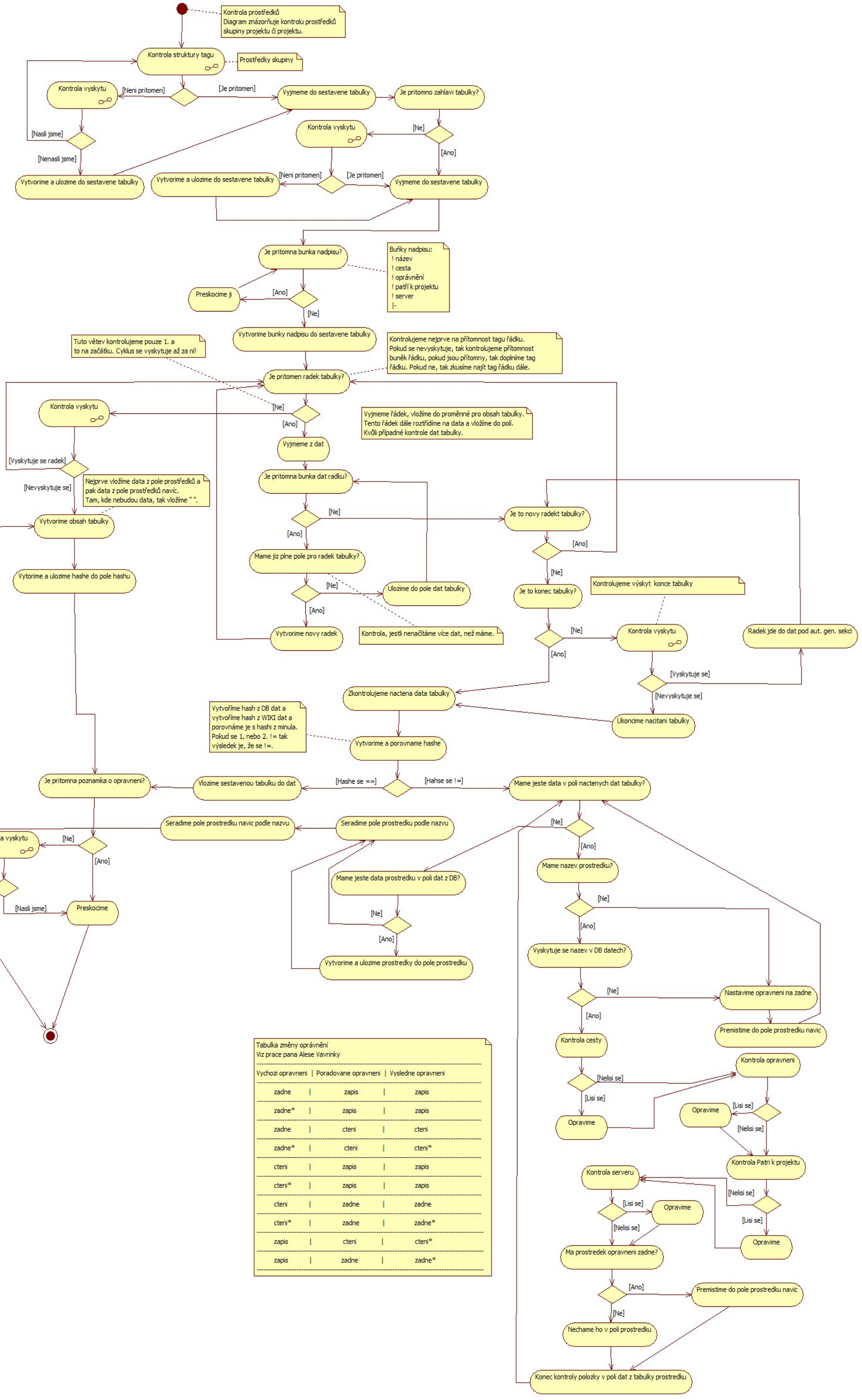


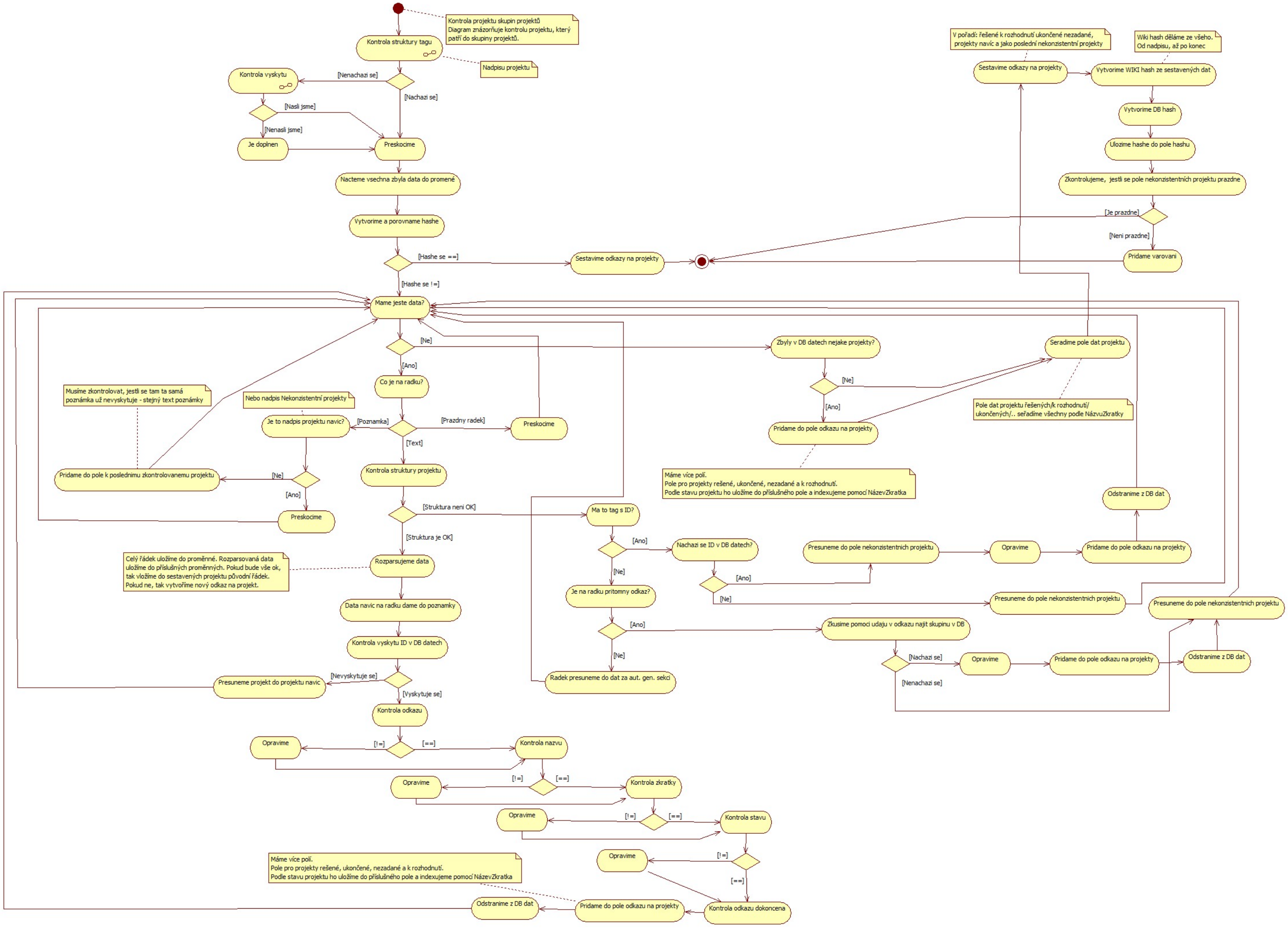

Vytvoření stránky skupiny projektu
Tento diagram zobrazuje vytvoření stránky pro skupin/y projektů.
Poznáme ji tak, že je prázdná.
Sestavíme z DB dat obsah stránky v požadovaném formátu do pole/proměnné. Pokud je více skupin na stránce, tak je nejprve seřadíme podle názvu a pak pomocí cyklu sestavíme.
Vytvoříme hashe pro každou skupinu a vložíme je do DB.
Pokud se něco pokazí, tak se stránka na WIKI nevloží a přidá se chyba.
Vytvoreni stranky
Pocet skupin v poli
[Mame >= 2 skupin]
[Mame 1 skupinu]
Vytvorime tag zacatku automaticky generovane sekce
Seradime pole podle nazvu skupin
Kontrola prazdnosti pole
[Pole je prazdne]
[Pole neni prazdne]
Vytvorime tag skupiny s ID
Sestavime 1 sekci s informacemi
Sestavime tabulku prostredku
Sestavime 3. sekci s projekty
Vytvorime tag konce skupiny
Vytvorime hashe
Pridame hash do pole Hashu
Vytvorime tag zacatku automaticky generovane sekce
Vytvorime tag skupiny s ID
Sestavime 1 sekci s informacemi
Sestavime tabulku prostredku
Sestavime 3. sekci s projekty
Vytvorime tag konce skupiny
Vytvorime tag konce automaticky generovane sekce
Vytvorime hashe
Pridame hash do pole Hashu
Vlozime stranku do WIKI
[addChyba(1) Nejde vytvorit stranka]
Chyba
Zalogujeme chybu
[Ok]
Začátek měření délky vytvoření skupiny projektů
Konec měření délky vytvoření skupiny projektů
Vytvorime tag konce automaticky generovane sekce


Kontrola vedouciho
Diagram znázorňuje kontrolu Vedouciho skupiny, či projektu
Kontrola struktury tagu
Kontrola vyskytu
[Vratil nenalezeni klicoveho slova]
[Vratil uspech]
Kontrola tagu s ID
[Nasli jsme]
[Nenasli jsme]
Vytvorime Vedouciho
[Je pritomen]
[Neni pritomen]
Porovname WIKI data s DB daty
Porovnáme, jestli se shoduje jméno a tituly umistěné ve WIKI a DB.
Kontrola jména a titulu
Porovname ho s DB hodnotou
[Vedouci se lisi]
V cem se lisi?
[V id, nebo jmene]
[Jen v titulu]
Porovname s hodnotou z tabulky hasheu
Uzivatel něco nezměnil.
Uděláme poznámku.
Vždy budeme mít pouze 1 poznámku
Vedouci doplneny uzivatelem.
bez tagu s ID.
[!=]
[==]
Pridame poznamku Vedouci doplneny uzivatelem
Uživatel nic nezměnil.
Pouze se změnila data v DB.
Jako ID nastavime NULL
[Nenalezli jsme]
[Nalezli jsme]
[Vedouci se shoduji]
Vytvorime hash a ulozime do pole hashu
Opravime titul
Opravime vedouciho
Doplnime ID
Kontrola jestli uz neni v predchozich
Pouze poud neni ID NULL,
tam by se zbytecne delala kontrola
2× po sobe.
Kontrolujeme jestli se tam nenáchazí stejné ID,
nebo pokud je tak místo ID: NULL, tak kontrolujeme
celé jméno i s titulem.
[Uz se tam vyskytuje]
[Neni tam jeste]
Kontrola aktualnosti dat
Preskocime predchozi vedouci a poznamky
Načítáme data, dokud máme předchozí vedoucí nebo poznámky.
Máme pole pro předchozí vedoucí, pole pro Vedoucí doplněné
uživatelem -> vedeme historii (krom případu, kdy jsou poznámky
naprosto totožné) a máme pole pro poznámky, které by uživatel mohl doplnit
[Data nejsou aktualni]
[Data jsou aktualni]
Opravime je
Vlozime data do prislusneho indexu pole skupin
Pridame ho do predchozich
Opravime vedouciho


Kontrola Redmine
Tento diagram znázorňuje kontrolu odkazu na redmine.
Kontrola struktury tagu
Kontrola vyskytu
[Neni pritomen]
[Je pritomen]
Kontrola vyskytu v DB datech
[Nasli jsme]
[Nenasli jsme]
Kontrola vyskytu v DB datech
[Nenachazi se]
[Nachazi se]
Zmenime hash na NULL
Vytvorime ho
[Nenachazi se]
[Nachazi se]
Kontrola dat
[Shoduji se]
[Neshoduji se]
Porovnani s hashem z minula
[Nelisi se]
[Lisi se]
Opravime data
Pridame do pole poznamek
Vytvorime a ulozime hash do pole hashu
Přidáme do pole poznámku
Uživatel ručne upravil Redmine:URL.
[Nenachazi se]
Nacteme minuly hash
Na stránce máme redmine.
V DB datech není (prazdné/NULL).
Při posledním spuštění skriptu něco na stránce bylo.
Kontrola, jestli to, co tam posledně bylo tam je i teď.
Pokud ANO => změna jen v DB a mažeme.
Pokud NE => uživatel něco připsal a děláme poznámku.
[Hash neni NULL]
[Hash je NULL]
Vytvorime hash
[Hashe se !=]
[Hashe se ==]
Zmenime hash na NULL
Vlozime to do poznamek
Vymazeme z hlavicky
Vymazeme z hlavicky
Na stránce máme redmine.
V DB datech není (prazdné/NULL).
Při posledním spuštění skriptu také nebyl.
=> Vymažeme s poznámkou
Uživatel ručně přidal Redmine:URL.


Kontrola Externí URL
Tento diagram znázorňuje kontrolu externí URL u projektů/ skupiny projektů
Kontrola struktury tagu
Kontrola vyskytu
[Neni pritomen]
[Je pritomen]
Kontrola vyskytu v DB datech
[Nasli jsme]
[Nenasli jsme]
Kontrola vyskytu v DB datech
[Nenachazi se]
[Nachazi se]
Zmenime hash na NULL
[Nenachazi se]
[Nachazi se]
Vytvorime ho
Vytvorime a ulozime hash do pole hashu
[Nenachazi se]
Nacteme minuly hash
[Nachazi se]
Kontrola dat
[Shoduji se]
[Neshoduji se]
Porovnani s hashem z minula
Opravime data
[Nelisi se]
[Lisi se]
Pridame do pole poznamek
Přidáme do pole poznámku
Uživatel ručně upravil Externí URL:URL.
Na stránce máme externí URL.
V DB datech není (NULL).
Při posledním spuštení skriptu něco na stránce bylo.
Kontrola, jestli to, co tam posledně bylo, tam je i teď.
Pokud ANO => změna jen v DB a mažeme.
Pokud NE => uživatel něco připsal a děláme poznámku.
[Hash neni NULL]
[Hash je NULL]
Vytvorime hash
[Hashe se !=]
[Hashe se ==]
Zmenime hash na NULL
Vlozime to do poznamek
Vymazeme z hlavicky
Vymazeme z hlavicky
Na stránce máme externí URL.
V DB datech není (NULL).
Při posledním spuštení skriptu také nebyl.
=> Vymažeme s poznámkou
Uživatel ručně přidal Externí URL:URL.


Kontrola prostředků
Diagram znázorňuje kontrolu prostředků skupiny projektu či projektu.
Kontrola struktury tagu
Prostředky skupiny
Kontrola vyskytu
[Neni pritomen]
[Je pritomen]
Vyjmeme do sestavene tabulky
Je pritomno zahlavi tabulky?
[Nasli jsme]
[Nenasli jsme]
Kontrola vyskytu
[Ne]
[Ano]
Vytvorime a ulozime do sestavene tabulky
Vytvorime a ulozime do sestavene tabulky
[Neni pritomen]
[Je pritomen]
Vyjmeme do sestavene tabulky
Je pritomna bunka nadpisu?
Buňky nadpisu:
! název
! cesta
! oprávnění
! patří k projektu
! server
|-
Preskocime ji
[Ano]
[Ne]
Vytvorime bunky nadpisu do sestavene tabulky
Tuto větev kontrolujeme pouze 1. a to na začátku. Cyklus se vyskytuje až za ní!
Kontrolujeme nejprve na přítomnost tagu řádku. Pokud se nevyskytuje, tak kontrolujeme přítomnost buněk řádku, pokud jsou přítomny, tak doplníme tag řádku. Pokud ne, tak zkusíme najít tag řádku dále.
Je pritomen radek tabulky?
Kontrola vyskytu
[Ne]
[Ano]
Vyjmeme řádek, vložíme do proměnné pro obsah tabulky. Tento řádek dále roztřídíme na data a vložíme do polí. Kvůli případné kontrole dat tabulky.
Vyjmeme z dat
[Vyskytuje se radek]
[Nevyskytuje se]
Nejprve vložíme data z pole prostředků a pak data z pole prostředků navíc. Tam, kde nebudou data, tak vložíme " ".
Je pritomna bunka dat radku?
[Ne]
[Ano]
Je to novy radekt tabulky?
[Ano]
Vytvorime obsah tabulky
Mame jiz plne pole pro radek tabulky?
[Ne]
[Ano]
Ulozime do pole dat tabulky
Kontrola, jestli nenačítáme více dat, než máme.
Vytvorime novy radek
[Ne]
Je to konec tabulky?
Kontrolujeme výskyt konce tabulky
[Ne]
[Ano]
Kontrola vyskytu
Radek jde do dat pod aut. gen. sekci
[Vyskytuje se]
[Nevyskytuje se]
Vytorime a ulozime hashe do pole hashu
Zkontrolujeme nactena data tabulky
Ukoncime nacitani tabulky
Vytvoříme hash z DB dat a vytvoříme hash z WIKI dat a porovnáme je s hashi z minula. Pokud se 1, nebo 2. != tak výsledek je, že se !=.
Vytvorime a porovname hashe
Je pritomna poznamka o opravneni?
Vlozime sestavenou tabulku do dat
[Hashe se ==]
[Hahse se !=]
Mame jeste data v poli nactenych dat tabulky?
Seradime pole prostredku navic podle nazvu
Seradime pole prostredku podle nazvu
[Ne]
[Ano]
Kontrola vyskytu
[Nenasli jsme]
[Nasli jsme]
Preskocime
Mame jeste data prostredku v poli dat z DB?
[Ne]
[Ano]
Vytvorime a ulozime prostredky do pole prostredku
Mame nazev prostredku?
[Ne]
[Ano]
Vyskytuje se nazev v DB datech?
[Ne]
[Ano]
Nastavime opravneni na zadne
Premistime do pole prostredku navic
Kontrola cesty
[Nelisi se]
[Lisi se]
Opravime
Kontrola opravneni
Opravime
[Lisi se]
[Nelisi se]
Kontrola Patri k projektu
Kontrola serveru
[Nelisi se]
[Lisi se]
Opravime
[Lisi se]
[Nelisi se]
Opravime
Ma prostredek opravneni zadne?
[Ano]
[Ne]
Premistime do pole prostredku navic
Nechame ho v poli prostredku
Konec kontroly polozky v poli dat z tabulky prostredku
Tabulka změny oprávnění
Viz prace pana Alese Vavrinky
Vychozi opravneni | Poradovane opravneni | Vysledne opravneni
zadne | zapis | zapis
zadne* | zapis | zapis
zadne | cteni | cteni
zadne* | cteni | cteni*
cteni | zapis | zapis
cteni* | zapis | zapis
cteni | zadne | zadne
cteni* | zadne | zadne*
zapis | cteni | cteni*
zapis | zadne | zadne*


Kontrola projektu skupin projektů
Diagram znázorňuje kontrolu projektu, který patří do skupiny projektů.
Kontrola struktury tagu
Nadpisu projektu
Kontrola vyskytu
[Nenachazi se]
[Nachazi se]
[Nasli jsme]
[Nenasli jsme]
Je doplnen
Preskocime
Nacteme vsechna zbyla data do promené
Vytvorime a porovname hashe
[Hashe se ==]
Sestavime odkazy na projekty
[Hashe se !=]
Mame jeste data?
[Ne]
[Ano]
Co je na radku?
Musíme zkontrolovat, jestli se tam ta samá poznámka už nevyskytuje - stejný text poznámky
Nebo nadpis Nekonzistentní projekty
Je to nadpis projektu navic?
[Poznamka]
[Prazdny radek]
Preskocime
[Text]
Pridame do pole k poslednimu zkontrolovanemu projektu
[Ne]
[Ano]
Preskocime
Kontrola struktury projektu
[Struktura neni OK]
[Struktura je OK]
Celý řádek uložíme do proměnné. Rozparsovaná data uložíme do příslušných proměnných. Pokud bude vše ok, tak vložíme do sestavených projektu původní řádek. Pokud ne, tak vytvoříme nový odkaz na projekt.
Rozparsujeme data
Data navic na radku dame do poznamky
Kontrola vyskytu ID v DB datech
[Nevyskytuje se]
Presuneme projekt do projektu navic
[Vyskytuje se]
Kontrola odkazu
Opravime
[!=]
[==]
Kontrola nazvu
Opravime
[!=]
[==]
Kontrola zkratky
Opravime
[!=]
[==]
Kontrola stavu
Opravime
[!=]
[==]
Kontrola odkazu dokoncena
Pridame do pole odkazu na projekty
Odstranime z DB dat
Máme více polí.
Pole pro projekty rešené, ukončené, nezadané a k rozhodnutí.
Podle stavu projektu ho uložíme do příslušného pole a indexujeme pomocí NázevZkratka
Ma to tag s ID?
[Ano]
[Ne]
Nachazi se ID v DB datech?
[Ano]
[Ne]
Je na radku pritomny odkaz?
[Ano]
[Ne]
Radek presuneme do dat za aut. gen. sekci
Zbyly v DB datech nejake projekty?
[Ne]
[Ano]
Pridame do pole odkazu na projekty
Máme více polí.
Pole pro projekty rešené, ukončené, nezadané a k rozhodnutí.
Podle stavu projektu ho uložíme do příslušného pole a indexujeme pomocí NázevZkratka
Seradime pole dat projektu
Pole dat projektu řešených/k rozhodnutí/ ukončených/.. seřadíme všechny podle NázvuZkratky
V pořadí: řešené k rozhodnutí ukončené nezadané, projekty navíc a jako poslední nekonzistentní projekty
Sestavime odkazy na projekty
Wiki hash děláme ze všeho. Od nadpisu, až po konec
Vytvorime WIKI hash ze sestavených dat
Vytvorime DB hash
Ulozime hashe do pole hashu
Zkontrolujeme, jestli se pole nekonzistentních projektu prazdne
[Je prazdne]
[Neni prazdne]
Pridame varovani
Odstranime z DB dat
Pridame do pole odkazu na projekty
Opravime
Presuneme do pole nekonzistentnich projektu
Nachazi se ID v DB datech?
[Ano]
[Ne]
Presuneme do pole nekonzistentnich projektu
Zkusime pomoci udaju v odkazu najit skupinu v DB
[Nachazi se]
[Nenachazi se]
Opravime
Pridame do pole odkazu na projekty
Odstranime z DB dat
Presuneme do pole nekonzistentnich projektu

A.5 Diagramy pro zpracování Kontroly projektů

Diagram modulu Projekty

Diagram Vytvoření stránky projektu

Diagram Kontroly řešitele

Diagram Kontroly řešení projektu

Diagram Kontroly typu projektu

Diagram Kontroly skupin projektu

Diagram Úkoly řešitelů k projektu

Diagram Výkazy k projektům

Struktura stránky projektu:
<!--NEGENEROVAT-->
__notitle____notoc__
<!-- Zacatek automaticky generovane sekce -->

<!--projekt:ID-->
Nazev(zkratka)

Vedouci: <!--vedouci:ID-->jmeno
:Předchozí vedouci: <!--idVedouciho:ID-->
:Předchozí vedouci: <!--idVedouciho:ID-->
Řešitel: <!--resitel:ID--> jmeno (resi....)
:Řešitel upravený uživatelem : xxxx
Zahájení projektu: datum pokud není řešitel, tak datum zadání
Ukončení řešení: pokud není, tak dáme -
Typ projektu: typ
Skupina projektu: skupina,skupina2, nebo -
:Poznámka, uživatel upravil skupiny na: data
Redmine: odkaz
Externi URL: url
Prostredky:
---
nazev	cesta	opravneni	patri k projektu	server
x	x	zadne	x	x
x	x	x	x	x
(Poznamka: K opravnění s * - dříve bylo oprávnění zápis)
Pracovní výkazy: odkaz
:Poznámky, uživatel přidal odkaz na výkazy
Ukoly: odkaz
:Poznámky, uživatel přidal odkaz na úkoly
....
<!--/projekt:ID-->

Hlavicky navic:
xxxxxx
<!-- Konec automaticky generovane sekce -->
Nekonzistentní hlavičky/:
(budou přesunuty pod tag aut. gen. sekce)
xxxx

Úprava knotis:
Přidáme pole pro externí URL do formuláře

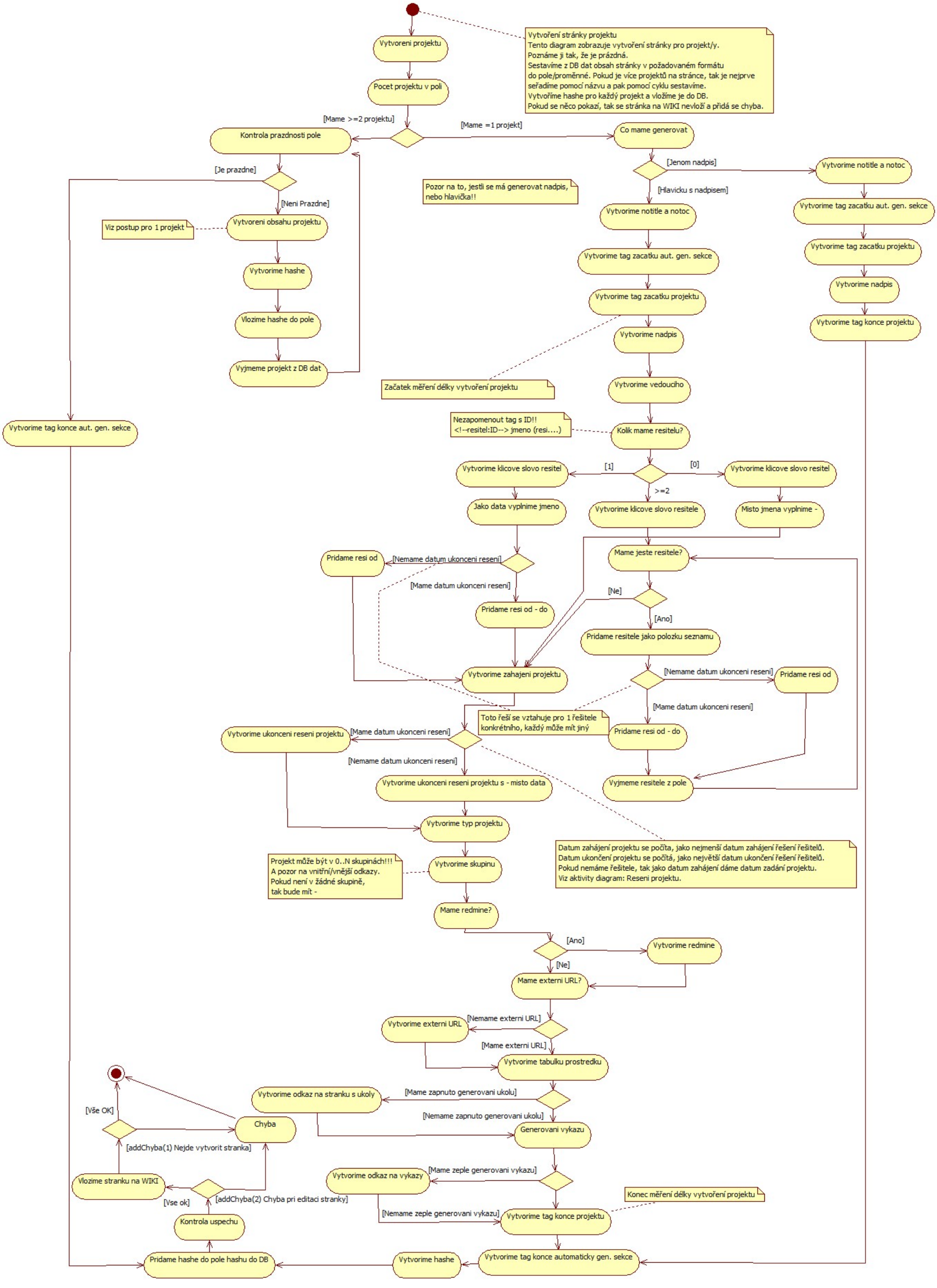
Úprava DB:
přidáme sloupec externí URL do tabulky.
Vytvoříme nový: generovani_wiki.
Přidáme nové sloupce do tabulky řeší:
zakazat_generovani_ukolu a zakazat_generovani_vykazu.

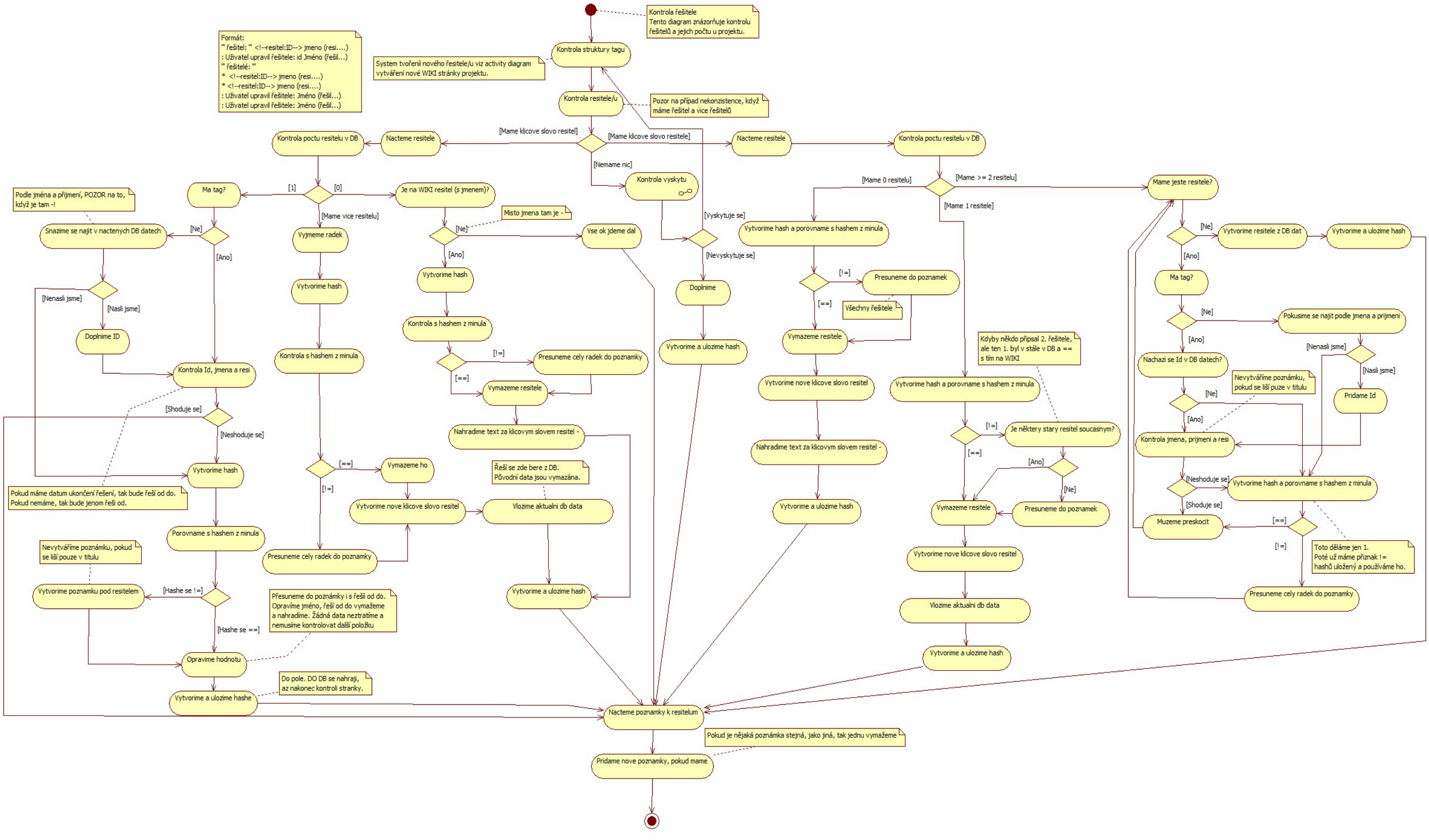
Vytvoříme tabulku projekty_hashes
<<PK>>projekt_ID
wiki_hash uděláme hash z wiki dat, včetně komentářů
DB_hash uděláme hash z DB dat, ať poznáme, jestli se od minule změnila hodnota v DB
nazev_projektu
vedouci - jméno+přijmení+tituly
resitel - bude hash, protože můžeme mít n rešitelů + reší od do
redmine
externi_url
hash_prostredku_wiki
hash_prostredku_db
typ - nebude hesh, ale nazev
skupina - hash
odkaz_na_ukoly hash odkazu
odkaz_na_vykazy hash odkazu na výkazy

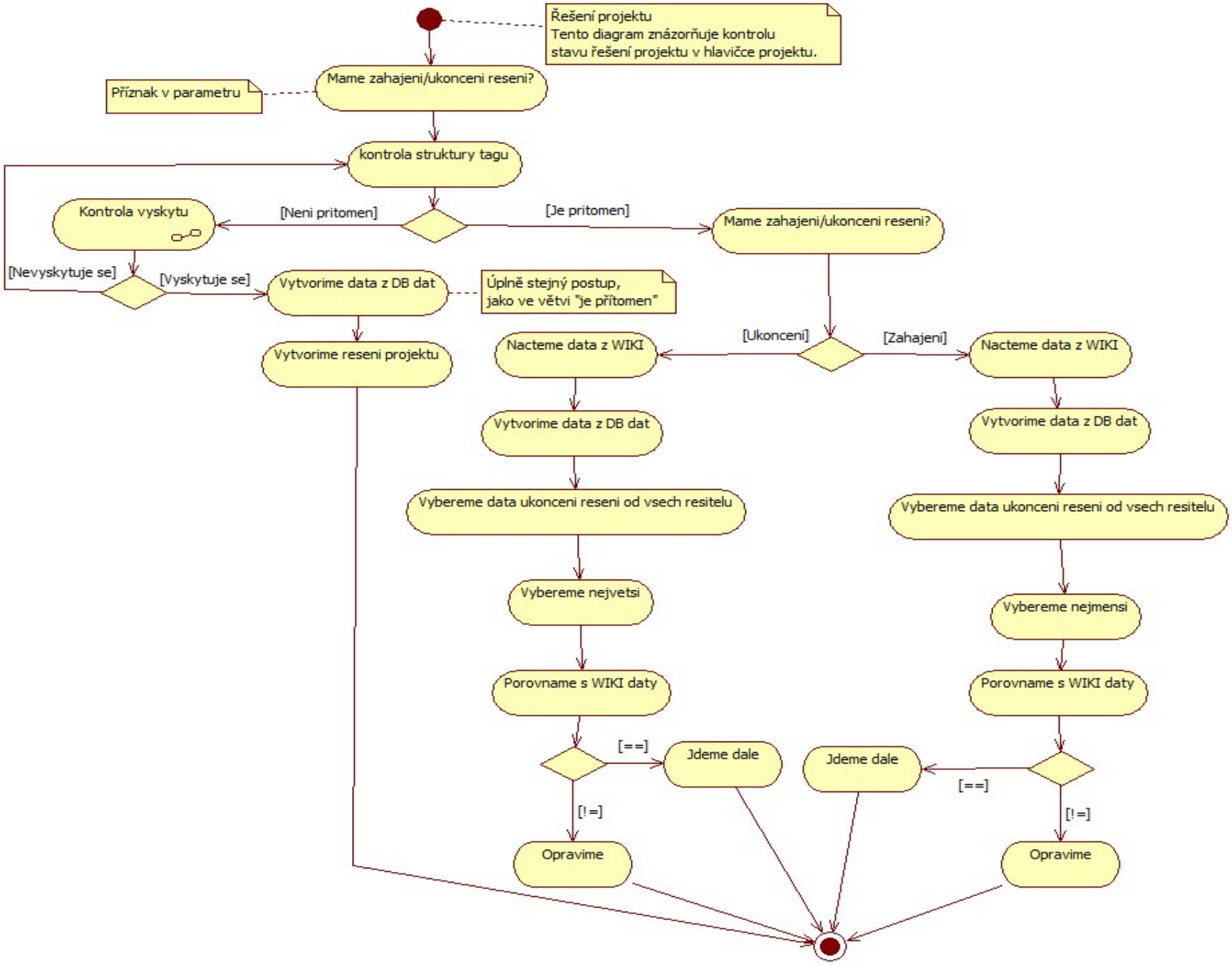
Projekt
Tento diagram zobrazuje kontrolu stránek projektů.
Na stránce může být 1..N hlaviček projektu a může se vyskytovat i hlavička skupiny.
V DB u projektu máme pole které nám určuje, co se bude generovat.
Rozšiřiřme KNOTIS o možnost zapnout/vypnout generování výkazu a úkolu.
Možnosti:
0: Negenerovat nic
1: Generovat nadpis
2: Generovat Hlavicku + Nadpis
3: Generovat Hlavicku + Napdpis + Ukoly
4: Generovat Hlavicku + Nadpis + Ukoly + Vykazy
U řešitele půjde explicitně vypnout generovaní úkolu/výkazu

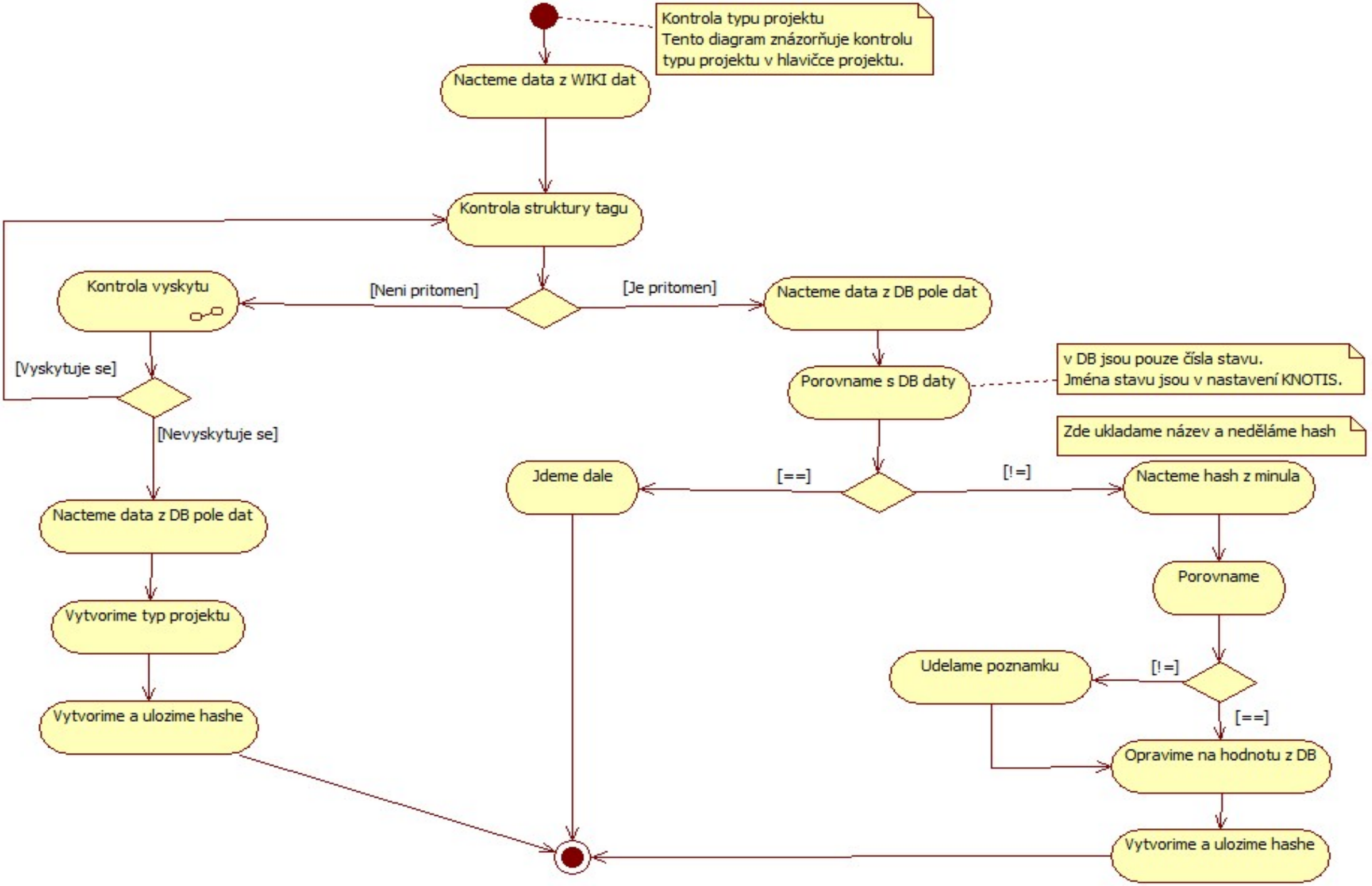
všech projektů, které mají generování >=1
SELECT id, url from projekt order by URL ASC
WHERE generovani..

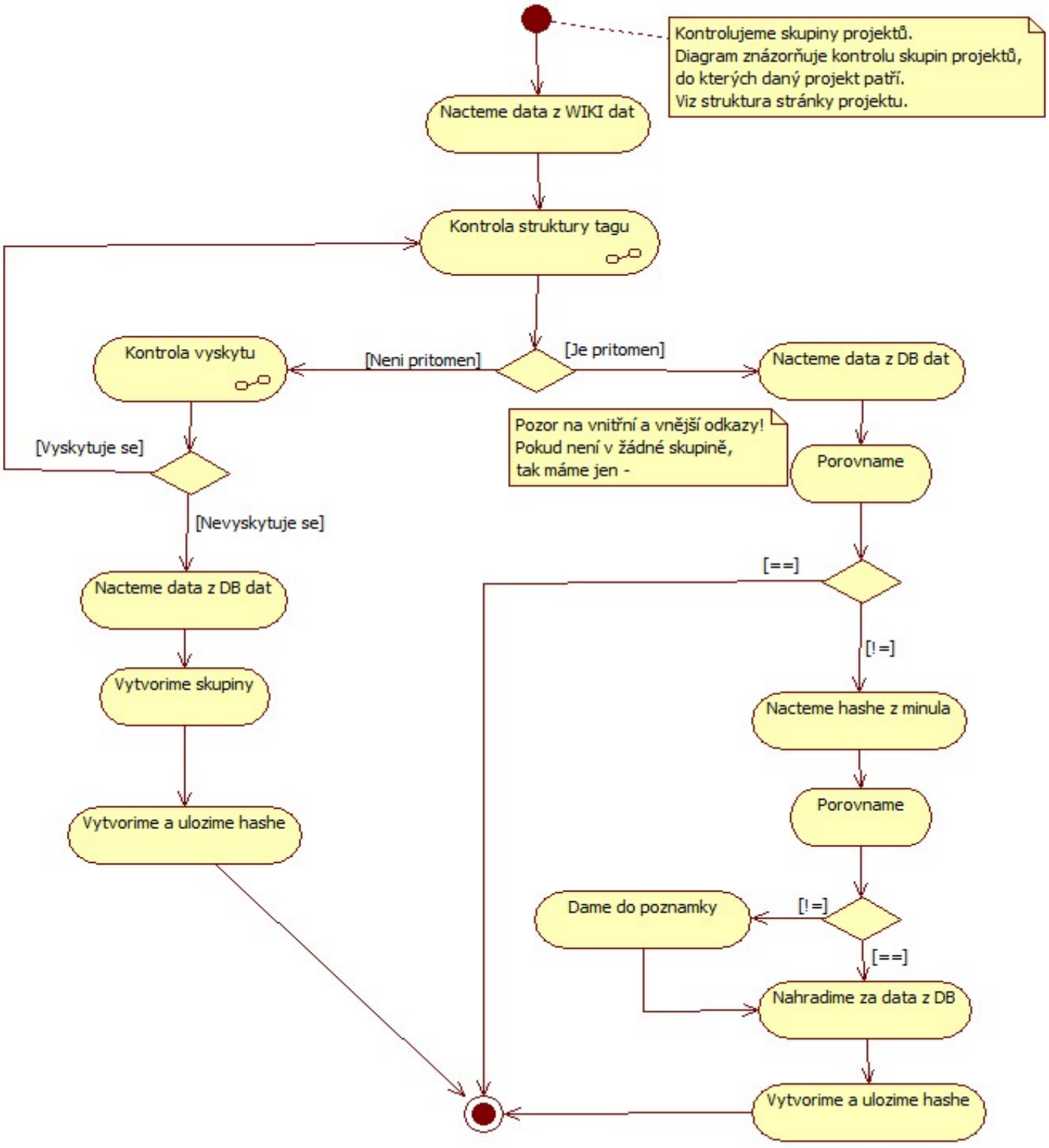
Nacteme id a URl vsech projektu
[Nepovedlo se]
[Povedlo se]
Seskupime je podle URL
Vyjmeme URL, ktere jsou ve vyjimkach
Kriticka chyba
Vybereme x seskupenych URL
X jde nastavit v nastavení programu
[Uz nemame dalsi URL]
Ukoncime statistiku pro projekty
[Mame jeste URL]
Nacteme podrobna data techto x projektu
Zalogujeme chybu
[Nenacetla se]
[Nacetla se]
Mame jeste URL v techto x nactenych URL?
[Ne]
[Ano]
Ulozime hashe do DB
[Podarilo se]
[Nepodarilo se]
Nacteme dalsi varku URL
Zalogujeme chybu
Editujeme WIKI stranku
Zkontrolujeme WIKI nahrady
Kontrola existence stranky
[Stranka existuje]
[Stranka neexistuje]
Editace stranky
Vytvor stranku projektu
[Uspesne vytvoreno]
[addChyba(2) Chyba pri editaci stranky]
Nacteme data z WIKI
addChyba(0) Chyba nacteni WIKI dat
Zalogujeme chybu
[Data se nacetla]
Je stranka prazdna?
[Ano]
[Ne]
Kontrola tagu negenerovat
[Je pritomen]
[Neni pritomen]
Kontrola zacatku stranky
Vlozime text zbytku stranky
Vlozime text, který ma byt pod aut. gen. sekci
Vlozime nekonzistentni hlavicky
Vytvorime tag konce aut. gen. sekce
Vlozime serazene hlavicky skupin
Vlozime hlavicky navic
Seradime skupiny, jestli mame
[Nebyly pritomny skupiny]
[Byly pritomny skupiny]
Vlozime serazene hlavicky projektu
Seradime pole projektu podle nazvu
Koncime s kontrolou na tomto url
[Ne]
[Ano]
Vlozime do pole skupin zbyle skupiny z DB dat
Vytvorime a ulozime hashe hlavicky do pole hashu
Je pole dat skupin z DB prazdne?
Mame na WIKI jeste data?
Pozor, když je na řádku poznámka, nebo nadpis hlavičky navíc
Vyjmeme prázdné řádky ze začátku dat.
Vezmeme 1. řádek a podíváme se, jestli se na něm nachází tag hlavičky.
Pokud ano, tak se podíváme, jestli máme i konec
Kontrola, jestli na radku je tag hlavicky
Identifikace tagu
Je to klicove slovo hlavicky navic?
Preskocime
[Skupina projektu]
Vyjmeme skupinu projektu
[Projekt]
Vyjmeme projekt
Projekt nemá tag konce.
Vyjmeme všechna data, co do něj patří.
Použijeme regExp. Poté projekt zkontrolujeme a případná chybějící data doplní kontrola projektu.
Poroz na to, že můžeme mít jen nadpis!!
Vyjmeme vsechna data, co patri do projektu
Vyjmeme vsechna data, co patri do skupiny
Nepodařilo se vyjmout skupinu (nemá tag konce)
Vyjmeme všechna data, která patří do skupiny.
Poté vytvoříme tag konce o zbytek se postará kontrola skupiny příště, pokud je tato skupina v DB.
Vyjmeme ji kvůli řazení hlaviček skupin a projektů.
A nastavíme ID NULL.
Pokud projekt nemá tag konce, tak pomocí regExpu vyjmeme všechna data, co do něj patří.
U projektů navíc neopravujeme, pokud mu chybí nějaké klíčové slovo. Jen se postaráme, aby měl tag začátku a konce.
Vyskytuje se id v poli dat?
Vyjmeme do pole projektu navic
[Id je NULL]
Pokusime se najit projekt v DB datech
Pokud mime ID:NULL, tak kontrolujeme podle názvu, zkratky a vedouciho
[Nasli jsme]
[Nenasli jsme]
Co mame vse generovat?
Vytvoříme odpovidající DB hash a porovnáme.
Načteme z DB DB hash poslední kontroly a wiki_hash poslední kontroly.
Porovnáme WIKI hash současný a WIKI hash poslední kontroly, pokud se != nastavíme příznak
Vytvořím DB hash a porovnám s DB hashem posledni kontroly, pokud se != nastavím příznak.
Výhoda, oproti predešlému způsobu je, ze můžeme poznat, jestli někdo neupravil něco v poznámce.
Pokud je nastaven alespon 1 příznak, tak se hashe !
[Jen nadpis]
[Celou hlavicku]
Vytvorime hashe a porovname, jen nadpisu
Vytvorime a porovname hashe
[==]
[!=]
Vyjmeme z DB dat
Vyjmeme do pole zkontrolovanyych projektu
Co mame na WIKI?
Reseni projektu
Kontrola resitele
Kontrola vedouciho projektu
Kontrola nadpisu
Kontrola nadpisu
Kontrola skupin projektu
Kontrola redmine
Kontrola externi URL
Kontrola typu projektu
Vykazy k projektum
Ukoly resitelu k projektu
Kontrola prostredku
Kontrola nadpisu
Kontrola pritomnosti uzivatelskych dat dale v hlavicce
Vymazeme zbytek hlavicky
[Nejsou pritomna]
[Jsou pritomna]
Preskocime data
Vytvorime a ulozime hashe hlavicky do pole hashu
Mame nastavenou promennou chybejiciho tagu konce aut. gen. sekce?
Je to nadpis?
Je to klicove slovo nekonzistentni hlavicky
Je to nadpis?
Je to projekt
Podle nadpisu a vedouciho
Vyjmeme vsechna data, co patri do projektu
Zkusime najit v DB datech
Co je na dalsim radku?
Pridame ID
Presuneme do pole nekonzistentnich hlavicek
Je to skupina projektu
Vyjmeme vsechna data, která patri do skupiny
Dame jim Id NULL
Vyjmeme skupinu
Je to nadpis?
Co je na dalsim radku?
Pokud je přítomen uživatelský nadpis, tak přeskočíme.
Presuneme text do dat za au. gen. sekci
Vlozime do dat za aut. gen. sekci
Vyjmeme data az do dalsiho klic. slova
Je do xx znaku pritomno klicove slovo?
I s načteným nadpisem
Vyjmeme data do klicoveho slova a dame do dat za aut. gen. sekci
Zbyla data vlozim do promenne pro zbyla data
[Tag vedouci]
[Tag vedouciho skupiny]
[Neco jineho]
[Vyskytuji se]
[Nevyskytuji se]

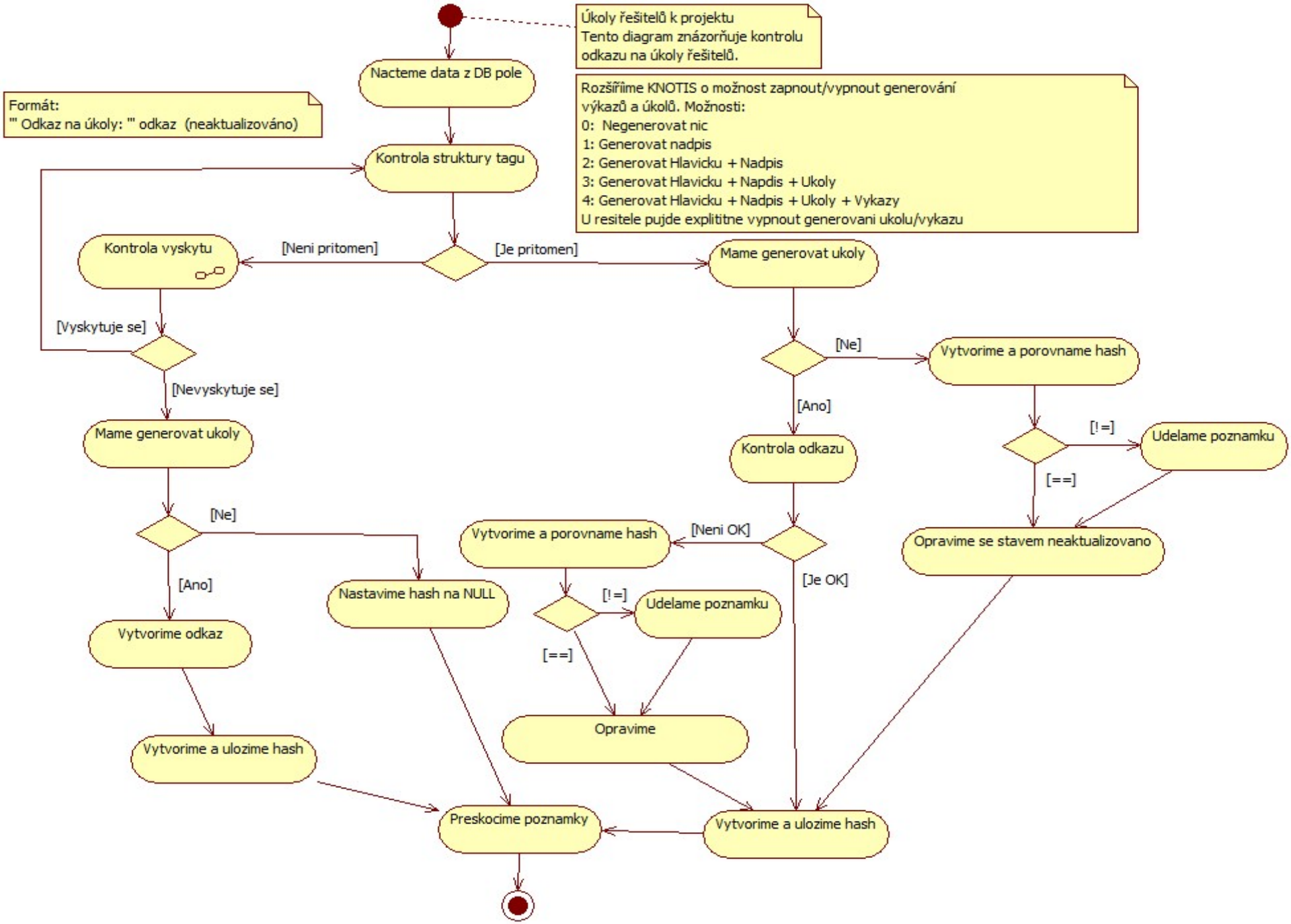

Vytvoreni projektu
Vytvoření stránky projektu
Tento diagram zobrazuje vytvoření stránky pro projekt/y.
Poznáme ji tak, že je prázdná.
Sestavíme z DB dat obsah stránky v požadovaném formátu
do pole/proměnné. Pokud je více projektů na stránce, tak je nejprve
seřadíme pomocí názvu a pak pomocí cyklu sestavíme.
Vytvoříme hashe pro každý projekt a vložíme je do DB.
Pokud se něco pokazí, tak se stránka na WIKI nevloží a přidá se chyba.
Pocet projektu v poli
[Mame >=2 projektu]
[Mame =1 projekt]
Kontrola prazdnosti pole
Co mame generovat
[Je prazdne]
[Neni Prazdne]
[Jenom nadpis]
[Hlavicku s nadpisem]
Pozor na to, jestli se má generovat nadpis,
nebo hlavička!!
Viz postup pro 1 projekt
Vytvoreni obsahu projektu
Vytvorime notitle a notoc
Vytvorime tag zacatku aut. gen. sekce
Vytvorime tag zacatku projektu
Vytvorime nadpis
Vytvorime tag konce projektu
Vytvorime hashe
Vlozime hashe do pole
Vyjmeme projekt z DB dat
Vytvorime notitle a notoc
Vytvorime tag zacatku aut. gen. sekce
Vytvorime tag zacatku projektu
Vytvorime nadpis
Začatek měření délky vytvoření projektu
Vytvorime vedouciho
Vytvorime tag konce aut. gen. sekce
Nezapomenout tag s ID!!
<!--resitel:ID--> jmeno (resi....)
Kolik mame resitelu?
[1]
[0]
>=2
Vytvorime klicove slovo resitel
Vytvorime klicove slovo resitel
Jako data vyplnime jmeno
Vytvorime klicove slovo resitele
Misto jmena vyplnime -
Pridame resi od
[Nemame datum ukonceni reseni]
[Mame datum ukonceni reseni]
Mame jeste resitele?
[Ne]
[Ano]
Pridame resi od - do
Pridame resitele jako polozku seznamu
Vytvorime zahajeni projektu
[Nemame datum ukonceni reseni]
Pridame resi od
[Mame datum ukonceni reseni]
Toto řeší se vztahuje pro 1 řešitele
konkrétního, každý může mít jiný
Vytvorime ukonceni reseni projektu
[Mame datum ukonceni reseni]
Pridame resi od - do
[Nemame datum ukonceni reseni]
Vytvorime ukonceni reseni projektu s - misto data
Vyjmeme resitele z pole
Vytvorime typ projektu
Datum zahájení projektu se počíta, jako nejmenší datum zahájení řešení řešitelů.
Datum ukončení projektu se počítá, jako největší datum ukončení řešení řešitelů.
Pokud nemáme řešitele, tak jako datum zahájení dáme datum zadání projektu.
Viz aktivity diagram: Reseni projektu.
Projekt může být v 0..N skupinách!!!
A pozor na vnitřní/vnější odkazy.
Pokud není v žádné skupině,
tak bude mít -
Vytvorime skupinu
Mame redmine?
[Ano]
[Ne]
Vytvorime redmine
Mame externi URL?
[Nemame externi URL]
Vytvorime externi URL
[Mame externi URL]
Vytvorime tabulku prostredku
Vytvorime odkaz na stranku s ukoly
[Mame zapnuto generovani ukolu]
[Nemame zapnuto generovani ukolu]
[Vše OK]
Chyba
Generovani vykazu
[addChyba(1) Nejde vytvorit stranka]
[Mame zeple generovani vykazu]
Vytvorime odkaz na vykazy
Vlozime stranku na WIKI
[Vse ok]
[addChyba(2) Chyba pri editaci stranky]
Konec měření délky vytvoření projektu
[Nemame zeple generovani vykazu]
Vytvorime tag konce projektu
Kontrola uspechu
Pridame hashe do pole hashu do DB
Vytvorime hashe
Vytvorime tag konce automaticky gen. sekce

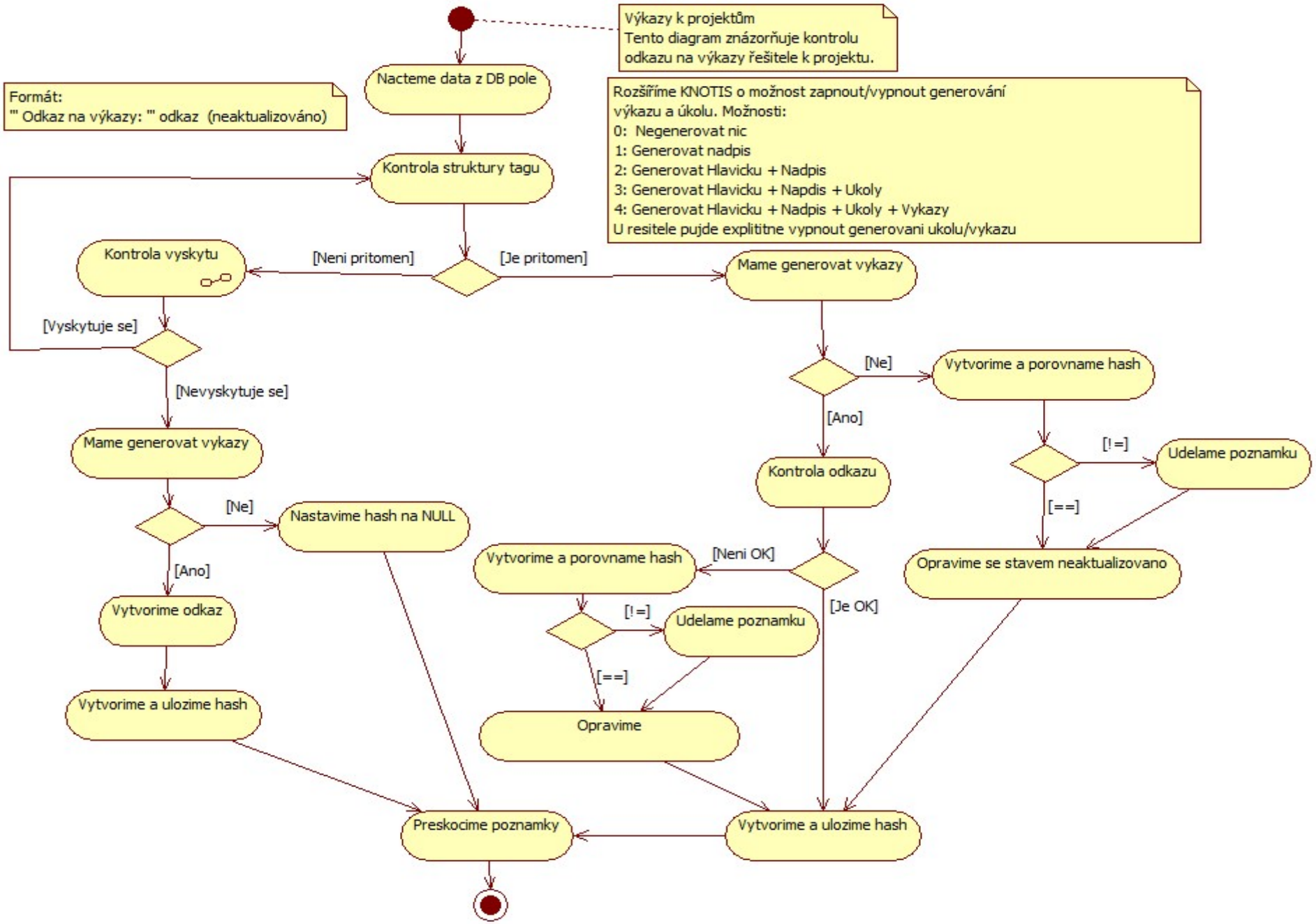

Kontrola řešitele
Tento diagram znázorňuje kontrolu řešitelů a jejich počtu u projektu.
Formát:
" řešitel: " <!--resitel:ID--> jmeno (resi....)
: Uživatel upravil řešitele: id Jméno (řešil...)
" řešitelé: "
* <!--resitel:ID--> jmeno (resi....)
* <!--resitel:ID--> jmeno (resi....)
: Uživatel upravil řešitele: Jméno (řešil...)
: Uživatel upravil řešitele: Jméno (řešil...)
System tvoření nového řesitele/u viz activity diagram vytváření nové WIKI stránky projektu.
Kontrola struktury tagu
Kontrola resitele/u
Pozor na případ nekonzistence, když máme řešitel a vice řešitelů
[Mame klicove slovo resitel]
[Mame klicove slovo resitele]
[Nemame nic]
Nacteme resitele
Kontrola poctu resitelu v DB
Kontrola vyskytu
[Vyskytuje se]
[Nevyskytuje se]
Nacteme resitele
Kontrola poctu resitelu v DB
Ma tag?
[1]
[0]
[Mame vice resitelu]
Je na WIKI resitel (s jmenem)?
Misto jmena tam je -
[Ne]
[Ano]
Vse ok jdeme dal
Podle jména a příjmení, POZOR na to, když je tam -!
Snazime se najit v nactenych DB datech
[Nenasli jsme]
[Nasli jsme]
Doplnime ID
Kontrola Id, jmena a resi
[Shoduje se]
[Neshoduje se]
Pokud máme datum ukončení řešení, tak bude řeší od do.
Pokud nemáme, tak bude jenom řeši od.
Vytvorime hash
Porovname s hashem z minula
Nevytváříme poznámku, pokud se liší pouze v titulu
Vytvorime poznamku pod resitelem
[Hashe se !=]
[Hashe se ==]
Opravime hodnotu
Vytvorime a ulozime hashe
Do pole. DO DB se nahraji, az nakonec kontroli stranky.
Vytvorime radek
Vytvorime hash
Kontrola s hashem z minula
[==]
[!=]
Vymazeme ho
Presuneme cely radek do poznamky
Vytvorime nove klicove slovo resitel
Řeší se zde bere z DB.
Původní data jsou vymazána.
Vlozime aktualni db data
Vytvorime a ulozime hash
Vytvorime hash
Kontrola s hashem z minula
[!=]
[==]
Presuneme cely radek do poznamky
Vymazeme resitele
Nahradime text za klicovym slovem resitel -
Vytvorime a ulozime hash
Doplnime
Vytvorime a ulozime hash
Vytvorime hash a porovname s hashem z minula
[!=]
[==]
Presuneme do poznamek
Všechny řešitele
Vymazeme resitele
Vytvorime nove klicove slovo resitel
Nahradime text za klicovym slovem resitel -
Vytvorime a ulozime hash
Kdyby někdo připsal 2. řešitele, ale ten 1. byl v stále v DB a == s tím na WIKI
Vytvorime hash a porovname s hashem z minula
[!=]
[==]
Je některy stary resitel soucasnym?
[Ano]
[Ne]
Vymazeme resitele
Presuneme do poznamek
Vytvorime nove klicove slovo resitel
Vlozime aktualni db data
Vytvorime a ulozime hash
[Mame 0 resitelu]
[Mame >= 2 resitelu]
[Mame 1 resitele]
Mame jeste resitele?
[Ne]
[Ano]
Vytvorime resitele z DB dat
Vytvorime a ulozime hash
Ma tag?
[Ne]
[Ano]
Pokusime se najit podle jmena a prijmeni
[Nenasli jsme]
[Nasli jsme]
Pridame Id
Nachazi se Id v DB datech?
[Ne]
[Ano]
Nevytváříme poznámku, pokud se liší puze v titulu
Kontrola jmena, prijmeni a resi
[Neshoduje se]
[Shoduje se]
Vytvorime hash a porovname s hashem z minula
Muzeme preskocit
[==]
[!=]
Toto děláme jen 1.
Poté už máme příznak != hashů uložený a používáme ho.
Presuneme cely radek do poznamky
Nacteme poznamky k resitelum
Pokud je nějaká poznámka stejná, jako jiná, tak jednu vymažeme
Pridame nove poznamky, pokud mame


Řešení projektu
Tento diagram znázorňuje kontrolu
stavu řešení projektu v hlavičce projektu.
Příznak v parametru
Mame zahajeni/ukonceni reseni?
kontrola struktury tagu
Kontrola vyskytu
[Neni pritomen]
[Je pritomen]
Mame zahajeni/ukonceni reseni?
[Nevyskytuje se]
[Vyskytuje se]
Vytvorime data z DB dat
Úplně stejný postup,
jako ve větvi "je přítomen"
Vytvorime reseni projektu
[Ukonceni]
[Zahajeni]
Nacteme data z WIKI
Nacteme data z WIKI
Vytvorime data z DB dat
Vytvorime data z DB dat
Vybereme data ukonceni reseni od vsech resitelu
Vybereme data ukonceni reseni od vsech resitelu
Vybereme nejvetsi
Vybereme nejmensi
Porovname s WIKI daty
Porovname s WIKI daty
[==]
Jdeme dale
Jdeme dale
[==]
[!=]
[!=]
Opravime
Opravime


Kontrola typu projektu
Tento diagram znázorňuje kontrolu typu projektu v hlavičce projektu.
Nacteme data z WIKI dat
Kontrola struktury tagu
Kontrola vyskytu
[Neni pritomen]
[Je pritomen]
Nacteme data z DB pole dat
[Vyskytuje se]
[Nevyskytuje se]
Porovname s DB daty
v DB jsou pouze čísla stavu.
Jména stavu jsou v nastavení KNOTIS.
Zde ukladame název a neděláme hash
Jdeme dale
[==]
[!=]
Nacteme hash z minula
Nacteme data z DB pole dat
Porovname
Vytvorime typ projektu
Udelame poznamku
[!=]
[==]
Vytvorime a ulozime hashe
Opravime na hodnotu z DB
Vytvorime a ulozime hashe


Kontrolujeme skupiny projektů.
Diagram znázorňuje kontrolu skupin projektů,
do kterých daný projekt patří.
Viz struktura stránky projektu.
Nacteme data z WIKI dat
Kontrola struktury tagu
Kontrola vyskytu
[Neni pritomen]
[Je pritomen]
Nacteme data z DB dat
Pozor na vnitřní a vnější odkazy!
Pokud není v žádné skupině,
tak máme jen -
Porovname
[Vyskytuje se]
[Nevyskytuje se]
[==]
Nacteme data z DB dat
[!=]
Vytvorime skupiny
Nacteme hashe z minula
Porovname
Vytvorime a ulozime hashe
Dame do poznamky
[!=]
[==]
Nahradime za data z DB
Vytvorime a ulozime hashe


Úkoly řešitelů k projektu
Tento diagram znázorňuje kontrolu odkazu na úkoly řešitelů.
Nacteme data z DB pole
Formát:
"' Odkaz na úkoly: "' odkaz (neaktualizováno)
Rozšíříme KNOTIS o možnost zapnout/vypnout generování výkazů a úkolů. Možnosti:
0: Negenerovat nic
1: Generovat nadpis
2: Generovat Hlavicku + Nadpis
3: Generovat Hlavicku + Napdis + Ukoly
4: Generovat Hlavicku + Nadpis + Ukoly + Vykazy
U resitele pujde explititne vypnout generovani ukolu/vykazu
Kontrola struktury tagu
Kontrola vyskytu
[Neni pritomen]
[Je pritomen]
Mame generovat ukoly
[Vyskytuje se]
[Nevyskytuje se]
Mame generovat ukoly
[Ne]
Vytvorime a porovname hash
[Ano]
Kontrola odkazu
[!=]
Udelame poznamku
[==]
[Ne]
Vytvorime a porovname hash
[Neni OK]
Opravime se stavem neaktualizovano
[Ano]
Nastavime hash na NULL
[!=]
Udelame poznamku
[Je OK]
Vytvorime odkaz
[==]
Opravime
Vytvorime a ulozime hash
Preskocime poznamky
Vytvorime a ulozime hash


Výkazy k projektům
Tento diagram znázorňuje kontrolu odkazu na výkazy řešitele k projektu.
Nacteme data z DB pole
Formát:
''' Odkaz na výkazy: ''' odkaz (neaktualizováno)
Kontrola struktury tagu
Rozšíříme KNOTIS o možnost zapnout/vypnout generování výkazu a úkolu. Možnosti:
0: Negenerovat nic
1: Generovat nadpis
2: Generovat Hlavicku + Nadpis
3: Generovat Hlavicku + Napdis + Ukoly
4: Generovat Hlavicku + Nadpis + Ukoly + Vykazy
U resitele pujde explititne vypnout generovani ukolu/vykazu
Kontrola vyskytu
[Neni pritomen]
[Je pritomen]
Mame generovat vykazy
[Vyskytuje se]
[Nevyskytuje se]
[Ne]
Vytvorime a porovname hash
[Ano]
[!=]
Udelame poznamku
Kontrola odkazu
[==]
Mame generovat vykazy
[Ne]
Nastavime hash na NULL
[Ano]
Vytvorime a porovname hash
[Neni OK]
Opravime se stavem neaktualizovano
Vytvorime odkaz
[!=]
Udelame poznamku
[Je OK]
[==]
Vytvorime a ulozime hash
Opravime
Preskocime poznamky
Vytvorime a ulozime hash

A.6 Diagramy pro Úkoly

Diagram modulu Úkoly

Diagram Vytvoření stránky úkolu

Diagram Kontrola vypracovat do

Úkol
Tento diagram znázorňuje kontrolu stránek úkolů.

Načteme si řešitele projektů, jehož projekty mají generování >= 3 a řešitel nemá zákaz generování. Seskupíme řešitele podle ID projektu. Potom vezmeme kus (jak velký kus bude v nastavení) projektů, načteme pro všechny řešitele v tom daném kusu projektů podrobné údaje a zkontrolujeme stránky úkolů těch projektů. Formát adresy: adresa_WIKI/index.php/ukoly_k_projektu/ID_projektu. Id projektu místo názvu/zkratky proto, protože se může název/zkratka změnit a předpokládám, že ID projektu je "trvanlivější" a jednoznačné. Poté načteme URL a zjistíme, jestli existuje/neexistuje. Atd.... Je to stejný princip, jako u kontroly projektů

Nacteme id resitelu a id projektu

[Nepovedlo se] → Kriticka chyba

[Povedlo se]

Seskupime resitele podle ID projektu

Vybereme x seskupenych projektu

X jde nastavit v nastavení programu

Zalogujeme chybu

[Nemame jiz projekty] → Ukoncime statistiku pro ukolu

[Mame jeste projekty]

Nacteme podrobna data pro resitele projektu

Editace stranky

[Nenacetla se]

[Nacetla se]

Nacteme data z WIKI

Mame jeste projekt v techto x nactenych projektech?

Kontrola zacatku stranky

[Ne] → Ulozime hashe do DB

[Ano]

Zkontrolujeme nazev projektu a ID

Kontrola existence stranky

[Podarilo se] → Nacteme dalsi varku projektu

[Nepodarilo se]

Zalogujeme chybu

[Nesouhlasi]

[Souhlasi]

[Stranka existuje]

[Stranka neexistuje] → Vytvoreni stranky Ukolu

[Uspesne vytvoreno]

[addChyba(2) Chyba pri editaci stranky]

Zalogujeme chybu

Mame jeste na WIKI data?

Opravime

[Ne] → Mame v Db datech jeste nejake resitele?

[Ano]

Mame hlavicku resitele?

[Ano] → Vytvorime a ulozime je do pole resitelu

[Ne]

[Ne]

Mame nastaveny priznak chybejiciho konce aut. gen. sekce?

[Ano]

[Ne] → Presuneme text pod aut. gen. sekci

Vyjmeme resitele

[Ano]

Mame dale v textu dalsi resitele?

[Ano]

[Ne]

Slozime stranku

Editujeme stranku na WIKI

[Nepovedlo se]

[Povedlo se]

Ulozime hashe do DB

[Povedlo se]

[Nepovedlo se]

Chyba

[Nepovedlo se]

[Povedlo se]

Vyjmeme vsechna data, co patri k resiteli

Vyskytuje se ID v DB?

[Ne]

Vyskytuje se v hlavicce neaktualizovano?

[Ano]

Kontrola jmena resitele

Zde pozor, může na řádku být (neaktualizováno)

[Ano] → Pridame do pole ukolu navic

[Ne]

Je v hlavicce uzivatelsky kontent?

[!=]

Vytvorime a porovname hash

[==]

Opravime

[!=]

Vytvorime poznamku

[Ne] → Vymazeme

[Ano]

Pridame Neaktualizovano

Mame jeste data hlavicky?

[Ne]

[Ano]

Mame v Db jeste ukoly pro resitele?

Mame tag ukolu?

[Ano]

Vytvorime je

[Ne]

[Ne]

Kontrola vyskytu

[Ano]

Je id v DB datech?

[Nasli jsme]

Konec kontroly resitele

Vezmeme data az po dalsi ukol a vlozimeje pod aut. gen. sekci

[Ne]

[Ano]

Vlozime data pod aut. gen. sekci

Nebo konec dat.
I když se úkol smaže v KNOTIS v DB zůstává se stavem smazáno!

Nacteme vsechna data co patri ukolu

[==]

[!=]

Kontrola poradi

Vytvorime a porovname hashe

Zadání může být na více řádků. Musíme tedy načítat řádky do té doby, než narazíme na poznámku, vypracovat do, nebo další element listu, nebo další tag úkolu

Vytvorime poznamku

[!=]

[==]

Kontrola textu zadani

Opravime

Vytvorime poznamku

[!=]

[==]

Kontrola vypracovat do

Zde pozor na text, který by tam mohl být navíc jen tak vložený

Mame tag stavu ukolu?

Opravime

Kontrola vyskytu

[Ne]

[Ano]

Je stav v DB?

Máme id?

Mame v Db stavy, které ve WIKI nejsou?

[Nenasli jsme]

[Nasli jsme]

[Ne]

Presuneme pod aut. gen. sekci

[Ano]

Konec kontroly ukolu

[Ano]

Doplnime a pridame do pole stavu

[Ne]

Vlozime data pod aut. gen. sekci

Kontrola dat stavu

Myšleno klasický, jaký je i v DB

Mame komentar ke stavu?

[==]

[!=]

Vytvorime poznamku

Je komentar v DB?

[Ano]

[Ne]

Opravime

Vyuzijeme hash

Kontrola textu komentare

[Ano]

[Ne]

Vytvorime poznamku

[!=]

[==]

Zmenime list na klasicky komentar :

Opravime

Konec kontroly stavu

Ulozime stav do pole stavu ukolu

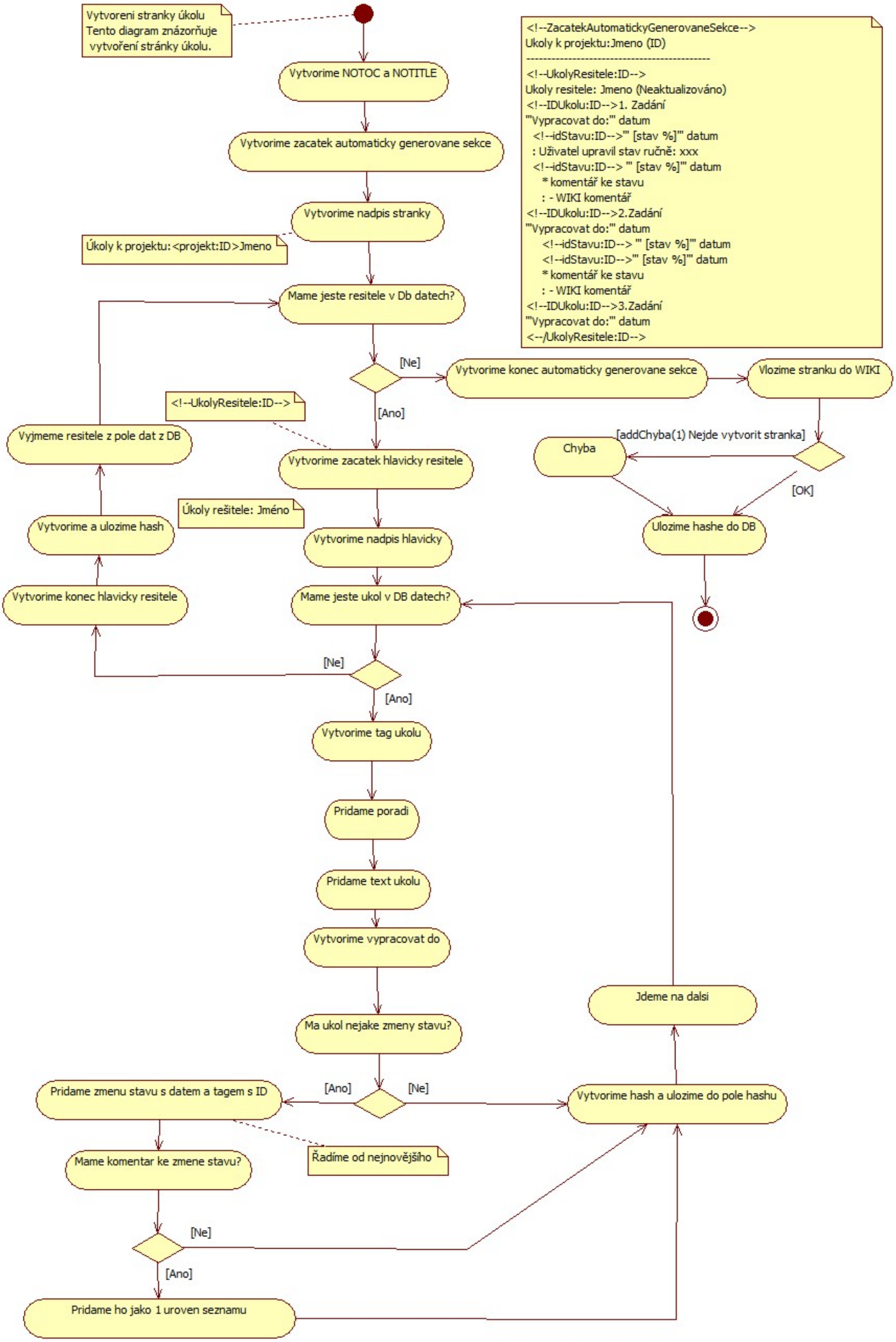
<!--ZacatekAutomatickyGenerovaneSekce-->
Ukoly k projektu:Jmeno (ID)

<!--UkolyResitele:ID-->
Ukoly resitele: Jmeno (Neaktualizováno)
<!--IDUkolu:ID-->1. Zadání
'"Vypracovat do:'" datum
<!--idStavu:ID-->'" [stav %]'" datum
: Uživatel upravil stav ručně: xxx
<!--idStavu:ID--> '" [stav %]'" datum
* komentář ke stavu
: - WIKI komentář
<!--IDUkolu:ID-->2.Zadání
'"Vypracovat do:'" datum
<!--idStavu:ID--> '" [stav %]'" datum
<!--idStavu:ID-->'" [stav %]'" datum
* komentář ke stavu
: - WIKI komentář
<!--IDUkolu:ID-->3.Zadání
'"Vypracovat do:'" datum
<--/UkolyResitele:ID-->

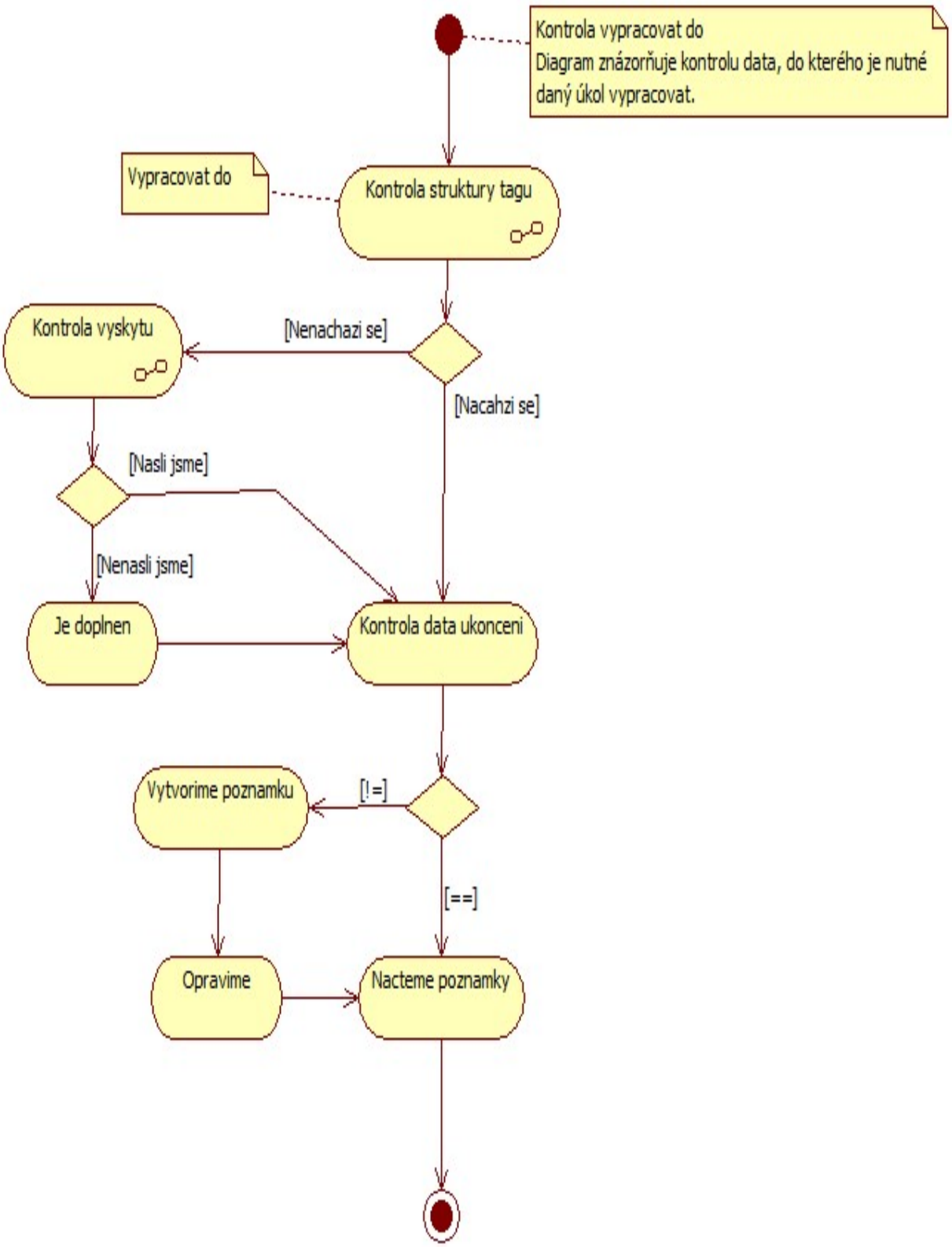
<!--UkolyResitele:ID2-->
Ukoly resitele: Jmeno2
.
.
.
<--/UkolyResitele:ID2-->
<--/AutomatickyGenerovanaSekce-->
Rozsirime KNOTIS o moznost zapnout/vypnout generovani vykazu a ukolu. Moznosti:
0: Negenerovat nic
1: Generovat nadpis
2: Generovat Hlavicku + Nadpis
3: Generovat Hlavicku + Napdis + Ukoly
4: Generovat Hlavicku + Nadpis + Ukoly + Vykazy
Úpravy tabulky řeší:
U resitele pujde explititne vypnout generovani ukolu/vykazu
zakaz_generovani_ukolu 1 -> znamená zákaz
zakaz_generovani_vykazu 1 -> znamená zákaz

Vytvoříme tabulku ukoly_hashes
id Je to stejne, jako id ukolu
wiki_hash hash wiki dat

Vytvoříme tabulku ukoly_resitel_hashes
id_resitele
wiki_hash


Vytvoreni stranky úkolu
Tento diagram znázorňuje
vytvoření stránky úkolu.
Vytvorime NOTOC a NOTITLE
Vytvorime zacatek automaticky generovane sekce
Vytvorime nadpis stranky
Úkoly k projektu:<projekt:ID>Jmeno
Mame jeste resitele v Db datech?
[Ne]
[Ano]
Vytvorime konec automaticky generovane sekce
Vlozime stranku do WIKI
<!--UkolyResitele:ID-->
Vyjmeme resitele z pole dat z DB
Vytvorime zacatek hlavicky resitele
[addChyba(1) Nejde vytvorit stranka]
Chyba
[OK]
Úkoly rešitele: Jméno
Vytvorime a ulozime hash
Vytvorime nadpis hlavicky
Ulozime hashe do DB
Vytvorime konec hlavicky resitele
Mame jeste ukol v DB datech?
[Ne]
[Ano]
Vytvorime tag ukolu
Pridame poradi
Pridame text ukolu
Vytvorime vypracovat do
Ma ukol nejake zmeny stavu?
Jdeme na dalsi
Pridame zmenu stavu s datem a tagem s ID
[Ano]
[Ne]
Vytvorime hash a ulozime do pole hashu
Řadíme od nejnovějšího
Mame komentar ke zmene stavu?
[Ne]
[Ano]
Pridame ho jako 1 uroven seznamu
<!--ZacatekAutomatickyGenerovaneSekce-->
Ukoly k projektu:Jmeno (ID)

<!--UkolyResitele:ID-->
Ukoly resitele: Jmeno (Neaktualizováno)
<!--IDUkolu:ID-->1. Zadání
'''Vypracovat do:''' datum
<!--idStavu:ID-->''' [stav %]''' datum
: Uživatel upravil stav ručně: xxx
<!--idStavu:ID--> ''' [stav %]''' datum
* komentář ke stavu
: - WIKI komentář
<!--IDUkolu:ID-->2.Zadání
'''Vypracovat do:''' datum
<!--idStavu:ID--> ''' [stav %]''' datum
<!--idStavu:ID-->''' [stav %]''' datum
* komentář ke stavu
: - WIKI komentář
<!--IDUkolu:ID-->3.Zadání
'''Vypracovat do:''' datum
<--/UkolyResitele:ID-->


Kontrola vypracovat do
Diagram znázorňuje kontrolu data, do kterého je nutné daný úkol vypracovat.
Vypracovat do
Kontrola struktury tagu
Kontrola vyskytu
[Nenachazi se]
[Nacahzi se]
[Nasli jsme]
[Nenasli jsme]
Je doplnen
Kontrola data ukonceni
Vytvorime poznamku
[!=]
[==]
Opravime
Nacteme poznamky

A.7 Diagramy pro Mazání projektu

Diagram Mazání projektu

Mazání projektu
Tento diagram znázorňuje proces mazání hlavicky projektu z WIKI.
Metoda se volá ve chvíli, kdy se admin rozhodne vymazat projekt ze systému KNOTIS.
Mazání projektu a hashů projektu z DB obstarává KNOTIS.
Vyjímka Exception se volá, pokud na stránce je nějaký uživatelský kontent.
Vyjímka Vyjimka se volá, pokud dojde k chybě kontroly/mazání.

Prihlasime s edo WIKI

[Neuspech]

[Uspech]

Existuje stranka na URL?

[Ne]

Vyvolame vyjimku Exception

[Ano]

Jsou na strance nejaka data?

[Ne]

Vymazame stranku

[Ano]

Povedlo se]

[Nepovedlo se]

Vyjmeme aut. gen. sekci

[Nemame]

Kontrolujeme, jestli mame data ve zbytku stranky

[Mame]

[Nemame]

Podivame se, jestli mame hlavicku hledaneho projektu

[Mame]

Vymazeme ze stranky aug. gen. sekci a ruseni nadpisu

[Nemame]

[Mame]

Vyjmeme projekt

Je aut. gen. sekce prazdna?

Vyvolame vyjimku Vyjimka

[Nepodarilo se]

[Neni]

[Podarilo se]

[Je]

Zjistime, jestli se v projektu nachazi uzivatelsky kontent

[Nachazi se]

[Nenachazi se]

Je po vyjmuti projektu aut. gen. sekce prazdna?

[Ano]

Kontrolujeme, jestli mame data ve zbytku stranky

[Ne]

[Nemame]

Slozime stranku dohromady

[Mame]

Editujeme stranku

Vymazeme ze stranky aug. gen. sekci a ruseni nadpisu

[Chyba]

[Uspech]

Odhlasime se z WIKI